Université de Montréal

DS-Fake: A data stream mining approach for fake news detection

par

Henri-Cedric Mputu Boleilanga

Département d'informatique et de recherche opérationnelle
Faculté des arts et des sciences

Mémoire présenté en vue de l'obtention du grade de
Maître ès sciences (M.Sc.)
en Discipline

August 15, 2022

Université de Montréal

Faculté des arts et des sciences

Ce mémoire intitulé

DS-Fake: A data stream mining approach for fake news detection

présenté par

Henri-Cedric Mputu Boleilanga

a été évalué par un jury composé des personnes suivantes :

Philippe Langlais
(président-rapporteur)

Esma Aimeur
(directeur de recherche)

Guy Lapalme
(membre du jury)

Résumé

L'avènement d'internet suivi des réseaux sociaux a permis un accès facile et une diffusion rapide de l'information par toute personne disposant d'une connexion internet. L'une des conséquences néfastes de cela est la propagation de fausses informations appelées «fake news». Les fake news représentent aujourd'hui un enjeu majeur au regard de ces conséquences. De nombreuses personnes affirment encore aujourd'hui que sans la diffusion massive de fake news sur Hillary Clinton lors de la campagne présidentielle de 2016, Donald Trump n'aurait peut-être pas été le vainqueur de cette élection. Le sujet de ce mémoire concerne donc la détection automatique des fake news.

De nos jours, il existe un grand nombre de travaux à ce sujet. La majorité des approches présentées se basent soit sur l'exploitation du contenu du texte d'entrée, soit sur le contexte social du texte ou encore sur un mélange entre ces deux types d'approches. Néanmoins, il existe très peu d'outils ou de systèmes efficaces qui détecte une fausse information dans la vie réelle, tout en incluant l'évolution de l'information au cours du temps. De plus, il y a un manque criant de systèmes conçues dans le but d'aider les utilisateurs des réseaux sociaux à adopter un comportement qui leur permettrait de détecter les fausses nouvelles.

Afin d'atténuer ce problème, nous proposons un système appelé DS-Fake. À notre connaissance, ce système est le premier à inclure l'exploration de flux de données. Un flux de données est une séquence infinie et dénombrable d'éléments et est utilisée pour représenter des données rendues disponibles au fil du temps. DS-Fake explore à la fois l'entrée et le contenu d'un flux de données. L'entrée est une publication sur Twitter donnée au système afin qu'il puisse déterminer si le tweet est digne de confiance. Le flux de données est extrait à l'aide de techniques d'extraction du contenu de sites Web. Le contenu reçu par ce flux est lié à l'entrée en termes de sujets ou d'entités nommées mentionnées dans le texte d'entrée. DS-Fake aide également les utilisateurs à développer de bons réflexes face à toute information qui se propage sur les réseaux sociaux.

DS-Fake attribue un score de crédibilité aux utilisateurs des réseaux sociaux. Ce score décrit la probabilité qu'un utilisateur puisse publier de fausses informations. La plupart des systèmes utilisent des caractéristiques comme le nombre de followers, la localisation, l'emploi, etc. Seuls quelques systèmes utilisent l'historique des publications précédentes d'un utilisateur afin d'attribuer un score. Pour déterminer ce score, la majorité des systèmes utilisent la moyenne. DS-Fake renvoie un pourcentage de confiance qui détermine la probabilité que l'entrée soit fiable. Contrairement au petit nombre de systèmes qui utilisent l'historique des publications en ne prenant pas en compte que les tweets précédents d'un utilisateur, DS-Fake calcule le score de crédibilité sur la base des tweets précédents de tous les utilisateurs. Nous avons renommé le score de crédibilité par score de légitimité. Ce dernier est basé sur la technique de la moyenne Bayésienne. Cette façon de calculer le score permet d'atténuer l'impact des résultats des publications précédentes en fonction du nombre de publications dans l'historique. Un utilisateur donné ayant un plus grand nombre de tweets dans son historique qu'un autre utilisateur, même si les tweets des deux sont tous vrais, le premier utilisateur est plus crédible que le second. Son score de légitimité sera donc plus élevé. À notre connaissance, ce travail est le premier qui utilise la moyenne Bayésienne basée sur l'historique de tweets de toutes les sources pour attribuer un score à chaque source.

De plus, les modules de DS-Fake ont la capacité d'encapsuler le résultat de deux tâches, à savoir la similarité de texte et l'inférence en langage naturel hl(en anglais Natural Language Inference). Ce type de modèle qui combine ces deux tâches de TAL est également nouveau pour la problématique de la détection des fake news. DS-Fake surpasse en termes de performance toutes les approches de l'état de l'art qui ont utilisé FakeNewsNet et qui se sont basées sur diverses métriques.

Il y a très peu d'ensembles de données complets avec une variété d'attributs, ce qui constitue un des défis de la recherche sur les fausses nouvelles. Shu *et al.* [**79**] ont introduit en 2018 l'ensemble de données FakeNewsNet pour résoudre ce problème. Le score de légitimité et les tweets récupérés ajoutent des attributs à l'ensemble de données FakeNewsNet.

Mots-clés : Détection de fausses nouvelles, Exploration de flux de données, IA explicable, score de légitimité, Traitement Automatique du Langage , Inférence du langage naturel, Similarité de texte, Reconnaissance d'entité nommée, Représentation d'encodeur bidirectionnel à partir d'un transformateur, Réseaux de neurones.

Abstract

The advent of the internet, followed by online social networks, has allowed easy access and rapid propagation of information by anyone with an internet connection. One of the harmful consequences of this is the spread of false information, which is well-known by the term "fake news". Fake news represent a major challenge due to their consequences. Some people still affirm that without the massive spread of fake news about Hillary Clinton during the 2016 presidential campaign, Donald Trump would not have been the winner of the 2016 United States presidential election. The subject of this thesis concerns the automatic detection of fake news.

Nowadays, there is a lot of research on this subject. The vast majority of the approaches presented in these works are based either on the exploitation of the input text content or the social context of the text or even on a mixture of these two types of approaches. Nevertheless, there are only a few practical tools or systems that detect false information in real life, and that includes the evolution of information over time. Moreover, no system yet offers an explanation to help social network users adopt a behaviour that will allow them to detect fake news.

In order to mitigate this problem, we propose a system called DS-Fake. To the best of our knowledge, this system is the first to include data stream mining. A data stream is a sequence of elements used to represent data elements over time. This system explores both the input and the contents of a data stream. The input is a post on Twitter given to the system that determines if the tweet can be trusted. The data stream is extracted using web scraping techniques. The content received by this flow is related to the input in terms of topics or named entities mentioned in the input text. This system also helps users develop good reflexes when faced with any information that spreads on social networks.

DS-Fake assigns a credibility score to users of social networks. This score describes how likely a user can publish false information. Most of the systems use features like the number of followers, the localization, the job title, etc. Only a few systems use the history

of a user's previous publications to assign a score. To determine this score, most systems use the average. DS-Fake returns a percentage of confidence that determines how likely the input is reliable. Unlike the small number of systems that use the publication history by taking into account only the previous tweets of a user, DS-Fake calculates the credibility score based on the previous tweets of all users. We renamed the credibility score legitimacy score. The latter is based on the Bayesian averaging technique. This way of calculating the score allows attenuating the impact of the results from previous posts according to the number of posts in the history. A user who has more tweets in his history than another user, even if the tweets of both are all true, the first user is more credible than the second. His legitimacy score will therefore be higher. To our knowledge, this work is the first that uses the Bayesian average based on the post history of all sources to assign a score to each source.

DS-Fake modules have the ability to encapsulate the output of two tasks, namely text similarity and natural language inference. This type of model that combines these two NLP tasks is also new for the problem of fake news detection.

There are very few complete datasets with a variety of attributes, which is one of the challenges of fake news research. Shu *et al.* introduce in 2018 the FakeNewsNet dataset to tackle this issue. Our work uses and enriches this dataset. The legitimacy score and the retrieved tweets from named entities mentioned in the input texts add features to the FakeNewsNet dataset. DS-Fake outperforms all state-of-the-art approaches that have used FakeNewsNet and that are based on various metrics.

keywords: Fake news detection, Data stream mining, Explainable AI, Legitimacy score, Natural Language Processing, Natural Language Inference, Text similarity, Named Entity Recognition, Bidirectional encoder representation from transformer, Neural Networks.

Contents

List of tables

List of figures

List of abbreviations and acronyms

BERT	*Bidirectional encoder representation from transformer*
BOW	*Bag of words*
CNN	*Convolutional Neural Network*
GNN	*Graph neural network*
GRUs	*Gated recurrent unit*
LSTM	*Long and short-term memory*
NLI	*Natural language inference*
RNN	*Recurrent neural networks*
SBERT	*Sentence - Bidirectional encoder representation from transformer*
SVM	*Support vector machine*
TF-IDF	*Term frequency–inverse document frequency*

Dédicace

À mon frère, Steve Wengi Boleilanga, j'espère que de là-haut tu es fier de moi. Je t'aime.

Acknowledgments

It's my great honour to take this opportunity to express my special thanks and gratitude to all the people who have contributed to achieving this work. Without their prayers, support and encouragement, I would not have been able to complete this thesis.

I would like to begin by thanking professor Esma Aïmeur who did me the honour of being my supervisor. All this would only be a child's dream without her invaluable advice, her unwavering support, her priceless guidance and her constructive criticisms during the past years. Once again, thank you for the person you are.

I also want to express my sincere gratitude to the jury members, Professors Philippe Langlais and Guy Lapalme, for taking their precious time to review and evaluate this thesis.

I am also indebted to all my teachers and classmates who crossed my path because, without each of them, I would not be the person I am. They spared no effort to help me in accomplishing this work. I particularly thank Professor Hicham Hage.

I want to express my deepest appreciation to my family, especially my dear parents, brothers, and sisters for always being there for me, for your sacrifices, for inspiring me on this journey and for your love in my life and my study. You are my strength, the reason why I get up every morning even more determined than the day before. Besides, I will thank my friends for all these moments of sharing and laughter that helped me get through difficult times.

Last, but not least, I would like to thank GOD.

Chapter 1

Introduction

This chapter presents an overview of our research work. It covers the problem, the motivations, the objectives and the contributions of this work. Finally, this chapter introduces the structure of this thesis.

1.1. Problem Definition

Humans have always sought to communicate, whether by using gestures, sounds, or inventing writing. The desire for communication eventually pushed human beings to create the internet. Over the years, the internet is taking a little more space in our lives.

This mix of the need to communicate and the fast and easy access to the internet no matter where you are in the world, mostly with the help of smartphones, led to the birth and the rapid growth of *online social networks*. Online social networks have thus given the possibility to anyone who wishes to express themselves and share information. Nowadays, the number of daily users of these social networks is in the millions. The world has become a small village where information circulates at high speed. Unfortunately, this does not only have good sides. We are increasingly identifying the effects of the harmful side of social networks. One of the most popular is known by the expression *"Fake news"*.

Authors, researchers or linguists haven't come yet with a unanimous definition of this expression. Mainly because the meaning and usage of this expression have changed over time. It was originally used to refer to false information and the dissemination of sensational information under the pretext of news reporting. The term has now evolved and becomes synonymous with the spread of false information [**74**]. Newspaper editors, back in the early 19th century, invented stories to increase the sale of their newspapers [**84**].

In 2019, the Oxford English Dictionary added the term as follows: "news that conveys or incorporates false, fabricated, or deliberately misleading information, or that is characterized as or accused of doing so" [**21**]. In scientific literature, many definitions have been presented. None of them is unanimous because they rely on the convergence and divergence of various related aspects of this phenomenon such as the type of information or the intention. For example, the following terminologies can be associated with the term Fake news [**68**]:

- Misinformation: distribute incorrect information accidentally, fooling its receivers;
- Malinformation: information that is mostly or completely true, but is disseminated with malice;
- Satire: a story presented as news that is factually incorrect, but the goal isn't to deceive, but to call out, ridicule, or expose shameful, corrupt, or otherwise bad behaviour;
- Propaganda: malinformation aimed at influencing a target audience and advancing a political objective;
- Hoax: a source of misinformation that can also be amusing;
- Spam: unwelcome material that clogs up the inboxes of those who receive it;
- Troll: disinformation used by a social media user to increase the tension between opposing beliefs;
- Infodemic: a mix of false and truthful information regarding a disease's origins and alternate therapies;
- Urban legends: a kind of folklore made consisting of rumours with supernatural, terrifying, or amusing components.

Thus, the problem addressed in this work is the detection of fake news on online social media. For the purposes of this thesis, inspired by the definition of Sharma *et al.* [**74**], we define the term "fake news" as follows.

Definition 1. Fake News: "A news article or message published and propagated through media, carrying false information regardless of the means and motives behind it".

1.2. Motivation and research goals

The term fake news experienced a surge in popularity around the time of the 2016 US presidential election, according to a Google Trends analysis. Since then, it is gaining momentum and influence our daily lives [**74**]. Corbu *et al.* [**17**] presented a survey that demonstrated that people's self-reported ability to spot fake news has a significant third-person effect and this effect is stronger when people compare their false news detection skills to that of distant others rather than close others. Fake news not only impacts us, but it spreads quickly and even faster than real news [**89, 74, 44**].

An important number of papers have been published on this problem. A wide variety of approaches have been tested, but few fake news detection systems perform well on a daily basis. Moreover, few works try to help users improve their fake news detection skills.

Nowadays, most recent approaches to detecting fake news rely on *machine learning* algorithms. However, such systems have a significant drawback when it comes to detecting fake news early on: the information needed to detect fake news is frequently absent or insufficient at the early stages of news dissemination. This is mainly the case for approaches that use the context for the detection. As a result, the accuracy of early detection has often been low. [**44**]

Nevertheless, some works have achieved good performance at an early stage, but the performance is strongly related to the used dataset since these types of approaches are mainly based on the *content mining* of the input. Few systems used external knowledge, such as Wikipedia [**3**]. This type of approach is limited since the information used to detect may not be reliable. In addition, we know that information evolves over time and Vosoughi *et al.* [**89**] showed that fake news propagates faster than real news. A piece of information can be true at a certain time and then be false shortly after. Thus, systems that use external data, like Wikipedia, where the information can not be trusted or when it is, it may take time to update it, can not be trustable tools for a real system that can be deployed to daily use.

Among the vast number of papers that address this problem, only a few focus on explaining the result of the used system to users that can help them realize their ability to

spot fake news and thus be able to improve it.

Motivated by the lack of powerful tools and the desire to help people, we decided to develop a system that responds to these issues. We named it *DS-Fake*. This name stands for *data stream* for fake news detection. Thus, DS-Fake is a *near-real-time* approach based on data stream mining. Near-real-time processing is when speed is crucial, but processing time in minutes is acceptable instead of seconds [**91**]. The term data mining defines the process of discovering interesting and useful patterns and relationships in large volumes of data [**16**]. Therefore, the term data stream mining defines the process of discovering interesting and useful patterns and relationships in large volumes of data that come from a flow of data. This flow of data allows DS-Fake to integrate the evolution of information over time. Thus, the system does not only process information that is available when a request is received but also processes future information that will be available in the near future after the request. To the best of our knowledge, this thesis is the first work that uses an approach based on data stream mining to detect fake news. This first version of DS-Fake works only on Twitter. It is built on the composition of four modules, each in charge of particular tasks. This system uses the input's content to receive data streams from trusted or legitimate sources. A data mining is then performed on both the input data and the broadcast data to determine the veracity of this input. Moreover, the context is also used in the calculation of the final percentage of confidence which is returned three times at three different time steps. At the end, one of the DS-Fake modules will generate an explanation for the user of this system.

The goal of this thesis can be summarized as follows. Building a near real-time system based on data stream mining that detects fake news. This system is based on a compromise between obtaining good performance at an early stage and the fact that the information evolves over time. On top of that, the system has to return an explanation that improves users' ability to spot fake news.

1.3. Contributions

The main contributions of this thesis are summarized as follows:

- The proposed system named DS-Fake is the first that uses a data stream mining approach for the fake news detection task to the best of our knowledge;

- This paper presents an explanation module that is a new variation of existing explanation methods;

- This work introduces the term *"legitimacy score"*. To our knowledge, this is the first time that this way of determining the score of a source has been used. It's associated with every Twitter user who has at least one post given to the system as input. DS-Fake computes a *percentage of confidence* that determines the probability that the input can be trusted by using the *Bayesian average* technique based on the history of the obtained percentage of posts from all sources;

- This work, to our knowledge, is the first on the fake news detection task that combines the result of two other natural language processing (NLP) tasks which are the text similarity and the natural language inference (NLI) task. This is the first time that the encapsulation of these tasks has been used for the fake news detection problem;

- The fake news detection task suffers from a lack of available datasets. And most of the time, when the dataset is available, it lacks essential features to perform a study that results in having a useful system [**74, 79**]. Shu *et al.* [**79**] presented a dataset named FakeNewsNet to mitigate this lack. We used this dataset and added features to it;

- The proposed system improves the *state-of-the-art* model results for the FakeNews-Net dataset.

1.4. Thesis organization

The thesis contains five chapters, including the first chapter, which is the introduction. The rest of this thesis is organized as follows:

- Chapter 2 presents the methods in the literature that address the fake news detection task and introduces the dataset used for our work;

- Chapter 3 presents the general architecture of the proposed system;

- Chapter 4 focuses on the experimentations made on the DS-Fake system. The used metrics are introduced first, then the rest of the chapter is a presentation and discussion of experiments;

- Chapter 5 discusses the conclusion and the future work of this thesis.

Chapter 2

Background and Related Work

In this chapter, we present a review of the literature on fake news detection. The second part of this chapter presents the dataset used during our experimentation and how our work contributes to enriching it with new features. This chapter shows how our proposed system contributes to the research on this problem. Among all the recent papers, none of them use a data stream mining approach or assign a score to sources using the Bayesian average technique. Additionally, none use the combination of techniques that we are going to present in the first part of this chapter. Moreover, this chapter reveals the absence of systems that includes an explainer.

2.1. Fake news detection

Sensational events have always sold well since the appearance of modern newspapers in the early 19th century. Newspaper makers were making fake stories to boost circulation [**84**]. Fake News in the present day is not the same as in the past. It belongs to a separate group from its historical counterparts because of the speed with which it spreads and the extent of its influence [**10**]. Despite the fact that the fake news detection problem was only just established, it has attracted a lot of attention. Various methods for detecting fake news in various forms of data have been proposed. This section contains a survey of existing and related works in the field of false news detection.

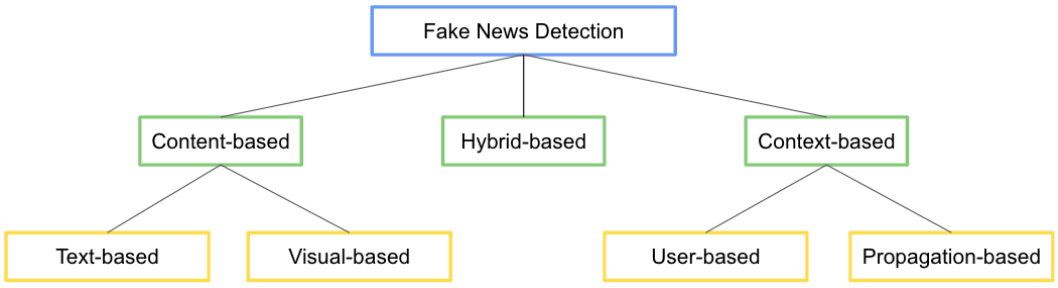


Fig. 2.1. Fake News detection approaches

Mahid *et al.* [**49**] distinguish, three families of automatic approaches that are used to detect fake news: the content-based approaches, the context-based approaches, and the hybrid-based approaches. These approaches can have sub-categories as shown in figure 2.1. In the following subsections, we are going to present those approaches. We will start by reviewing the recent advancements in the content-based method, then the social content-based method and we will finish with the mixing method.

2.1.1. Content-based approaches

The content-based family regroups all the approaches mainly based on data directly extracted from the actual content of the input which can be a text, a video, etc. As we saw in figure 2.1, this group of approaches can be divided into two subgroups: text-based approaches and visual-based approaches [**49**].

The text-based approach utilizes linguistic-based features that are extracted from text content at the level of characters, words, phrases, and documents. Figure 2.2 shows the type of content for a text-based approach. Approaches in this subgroup use cues that range from the lexical level to the discourse level. This sub-group contains text representation approaches, style-based approaches and knowledge-based approaches [**49**].

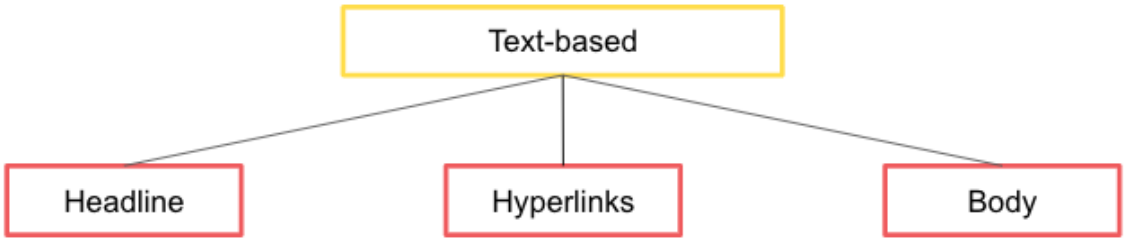


Fig. 2.2. Type of content for a text-based approach

The text representation approach uses text representation like n-grams, bag-of word, part-of-speech, etc. Another type of this approach is to train a model to detect if an input text is a fake-news by using content features such as the number of words, the percentage of stop-words, the length of the title, the number of special characters, the readability, the extracted sentiments, etc. The style-based approach uses cues in a text to detect fake news by considering the writing style of the news content. The usage of a knowledge-based approach is utilized to verify the accuracy of statements in a news article and is seen as a more scalable fact-checking tool. By accessing external resources such as existing knowledge networks or publicly available structured data, a deception of a particular data that is represented in the form of false "factual claims" can be extracted and reviewed to assess its truthfulness [**49**].

The second sub-method is the visual-based approach. It relies on various characteristics of distortion collected by the visual-based features extracted from images, videos or graphic interchange format (GIF), as we can see in figure 2.3. [**49**]

One of the first modern automatic approaches using the content-based detection models was developed by Ott *et al.* [**56**]. They used n-gram term frequency with an SVM classifier to detect fake opinions. They created a "gold-standard" dataset by collecting honest opinions from TripAdvisor and added deceptive opinions of hotels created with the help of Amazon Mechanical Turk.

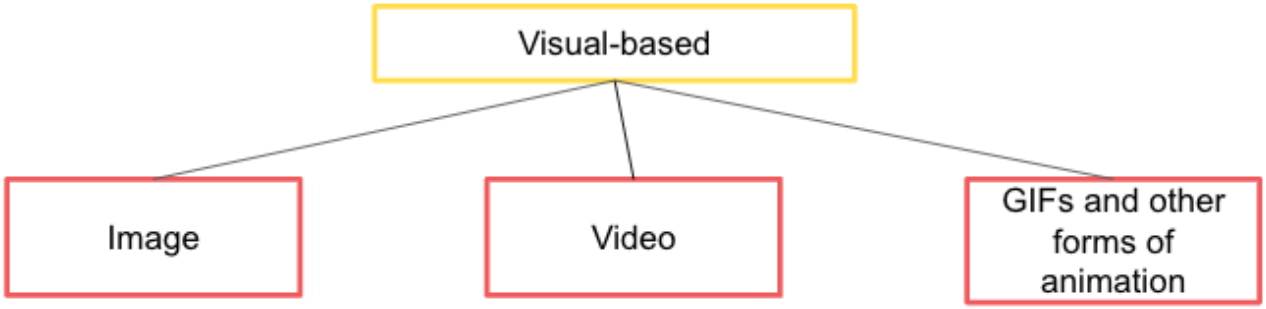


Fig. 2.3. Type of content for a visual-based approach

Rusli *et al.* [**69**] also used the n-gram model. Their approach was to make a comparison between the TF-IDF model and the Bag of Words model to extract features along with the use of the n-gram model. They implemented a supervised machine learning approach using the Multi-Layer Perceptron (MLP) to detect fake news articles on a dataset of 600 news articles written in Bahasa Indonesia that were acquired from a variety of web sources.

Since the end of the year 2019, our life started changing with the coronavirus. A lot of information circulated about the virus, the vaccine, etc. Obviously, false information also spread. Patwa *et al.* [**57**] collected and annotated 10700 social media posts and articles of real and fake news on COVID-19. They used 4 machine learning approaches baselines on this dataset to determine whatever a post or an article is real: Decision Tree; Logistic Regression (LR); Gradient Boost; and Support Vector Machine (SVM).

The same dataset was used by Wani *et al.* [**90**]. The authors used the following algorithms: Convolutional Neural Networks (CNN); Long Short Term Memory (LSTM, Bi-LSTM + Attention, Hierarchical Attention Networks HAN.); Bidirectional Encoder Representations from Transformers (BERT, DistilBERT).

Nasir *et al.* [**53**] presented a new hybrid deep learning model for detecting fake news that blends a convolutional neural network CNN and a recurrent neural network RNN as shown in figure 2.4. This combination of CNN-RNN was proposed because it can capture both local and sequential properties of input data. It has been proven successful in a variety of classification and regression applications.

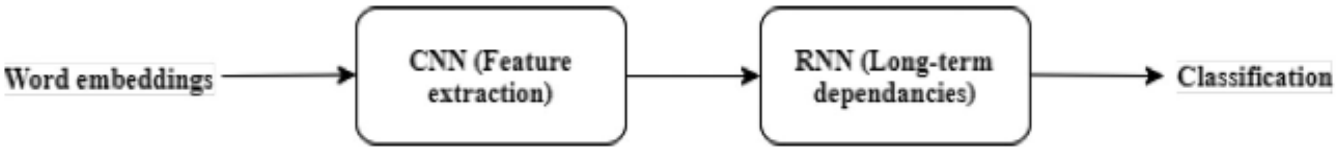


Fig. 2.4. The hybrid model proposed by Nasir *et al.* [**53**].

In September 2021, The Iberian Languages Evaluation Forum (IberLEF) organized the second edition of the FakeDeS 2021. The FakeDeS shared task intends to investigate various approaches and tactics for detecting fake news in Spanish. A new corpus that includes news about COVID-19, as well as news from other Ibero-American countries, was presented for this purpose. We will present the result of some interesting approaches that were suggested.

Spalenza *et al.*[**86**] proposed an approach that applies Machine Learning using Named Entity Recognition (NER) in addition to Part-Of-Speech tag (POSTag) sequences. The authors' strategy was to identify language models for factual and non-factual articles, as well as to recognize certain linguistic patterns. Through writing patterns, language modelling seeks to discover fake and true classes. The goal was to create a system that can recognize and learn to evaluate writing coherence. They were expecting incoherent, prejudiced, repetitious, and inaccurate linguistic formats in fake classified articles. So, they used POSTag+NER to organize the data in vectors of linguistic patterns. After that, they tested four classifiers to analyze the linguistic modelling from different perspectives: SVM; Random Forest (RF); Gradient Boosting (GB); Wilkie, Stonham and Aleksander's Recognition Device (WiSARD).

The analysis of text features was another approach presented during the forum. Reyes-Magana *et al.* [**66**] propose to determine whether a news article is fake by analyzing its textual representation. Logistic Regression (LR), Support Vector Machines (SVM) and Multinomial Naive-Bayes (NB) are the classifiers that were tested.

The last approach that we will present related to the second edition of the FakeDeS 2021 was presented by Luo [**48**]. The author used an unsupervised language representation learning technique based on a large corpus to learn a good feature representation for words called ALBERT. It uses fewer model parameters than BERT or OpenAI Generative Pre-trained Transformer (GPT) due to techniques such as parameter sharing and matrix decomposition.

Przybyła [**59**] found that it is a worthwhile direction of research. The author arrived at this conclusion based on the higher classification accuracy obtained during his experiments. The author also tried to predict whether an article is fake based on the style they are written in. Przybyła [**59**] created two new classifiers: a neural network (BiLSTMAvg) and a stylometric features-based model. BiLSTMAvg is a neural network with architecture based on elements used in natural language processing (NLP) such as word embeddings and bidirectional LSTM. BiLSTMAvg stands for bidirectional LSTM Average. To compute the score of an article, the author computes the average of all sentences of that article. The stylometric features-based model consists of a collection of stylistic features like the average sentence length and the average word length. The results show that the suggested classifiers maintain a relatively good accuracy when dealing with papers on previously unknown topics (e.g., new events) and from unknown sources (e.g. emerging news websites). An analysis of the stylometric model indicates it indeed focuses on sensational and affective vocabulary, known to be typical for fake news.

A theory-driven model that examines news content by capturing its writing style on four language levels was presented by Zhou *et al.* [**98**]. The different levels are lexicon-level, syntax-level, semantic-level, and discourse-level. They tried the following supervised classifiers: Logistic Regression (LR), Naïve Bayes (NB), Support Vector Machine (SVM), Random Forests (RF), and XGBoost. PolitiFact and BuzzFeed were the two datasets used by the authors. Figure 2.5 presents an example of the writing style captured from the following post at a discourse-level:

"Huffington Post is really running with this story from The Washington Post about the CIA confirming Russian interference in the presidential election. They're saying if 100% true, the courts can put Hillary in The White House!"

.

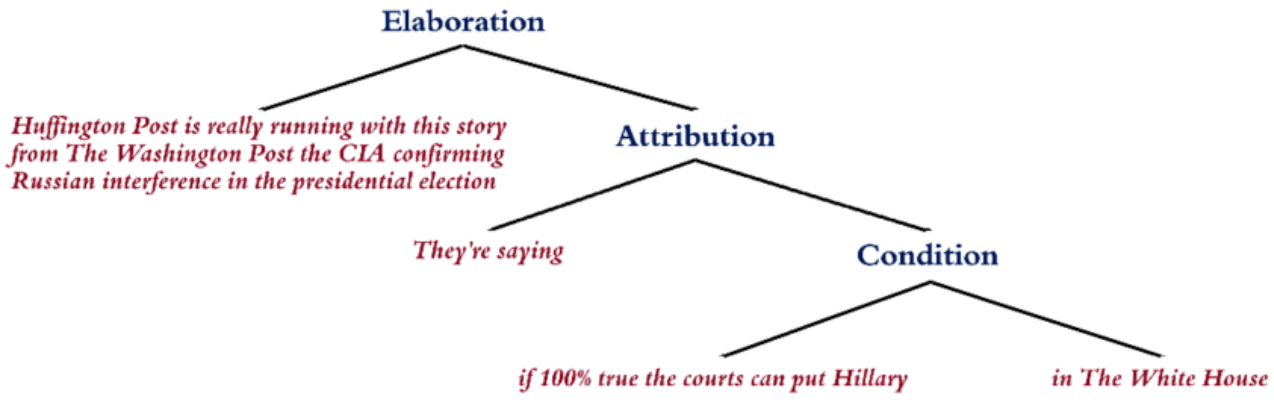


Fig. 2.5. A writing style example at discourse-level

To address limitations of computational methods for detecting fake news based on news content such as encoding semantics from original texts limited to the structure of the language in the text, making both bag-of-words and embedding-based features deceptive in the representation of fake news, and explainable methods frequently overlook relational contexts in fake news detection, Wu *et al.* [**92**] presented a new framework named knowledge graph enhanced KGF.

KGF is a knowledge graph framework system that can be applied in a variety of deep learning models for detecting fake news and presenting relational explanations [**92**]. By extracting entity relation tuples from the training data, they first build a credential-based multi-relation knowledge graph and then used a compositional graph convolutional network to learn the node and relation embeddings. The pre-trained graph embeddings are then incorporated into a graph convolutional network for fake news detection. During this study, three datasets were used: Celebrity, PolitiFact and GossipCop dataset.

So far, all the approaches that we saw were based on binary classification. During the 4th edition of the lab CheckThat! at CLEF 2021, they shared a new dataset of news articles with four possible labels for each article. The possible labels are false information; partially false; true; other. Those articles were collected on the sites of several fact-checking sites and were analyzed. This conference took place from September 21 to 24, 2021. We will present some interesting model that was presented during the CLEF2021-CheckThat!

The first place on the validation set goes to Kumari [**40**]. The author used BERT based classification model to predict the labels of an article. External data from the fact-checked articles were downloaded and added to the dataset provided by CLEF2021-CheckThat! to train the model.

Author	Used technique	Publish date
Canhasi *et al.* [**11**]	Machine learning (SVM, XGBoost, etc.)	2022
Gupta *et al.* [**31**]	Deep learning (RoBERTa)	2022
Refaeli *et al.* [**63**]	Deep learning (BERT)	2022
Harrag *et al.* [**33**]	Deep learning (CNN)	2022
Singhal *et al.* [**82**]	Deep learning (CNN)	2022
Xua *et al.* [**95**]	Machine learning (Graph)	2022
Karnyoto *et al.* [**38**]	Deep learning (Graph convolutional networks)	2022
Flores *et al.* [**25**]	Deep learning (BERT - Adversarial Attacks)	2022
Jain *et al.* [**36**]	Machine learning (Random forest)	2021
Hasan *et al.* [**34**]	Deep learning (LSTM, Ensemble methods)	2021
Bamiro *et al.* [**4**]	Machine learning (Random forest)	2021
Bekoulis *et al.* [**6**]	Deep learning (GNN)	2021
Gôlo *et al.* [**28**]	Deep learning (VAE)	2021
Fawaid *et al.* [**24**]	Deep learning (BERT with Transformer Network)	2021
De *et al.* [**20**]	Deep learning (BERT)	2021

Table 2.1. A list of recent papers on Fake news detection based on the content

An approach that uses data augmentation was proposed by Ashraf *et al.* [**2**]. This method consists of the insertion of alternative words. Word2vec embeddings were used to add a substitute for sentences. They used n-grams with several machine learning classifiers such as logistic regression, multilayer perceptron, support vector machine and random forest.

They report the result of each classifier with and without the augmented dataset. The result shows that the insertion of alternative words was not beneficial for this classification task.

The last paper that we are going to discuss related to the content-based approaches is the paper from Pritzkau [**58**]. The author presented a multi-class classification approach to detect fake news in texts. RoBERTa (A Robustly Optimized BERT Pretraining Approach) was used as a neural network architecture for sequence classification to assign class labels to the given texts. The author started with a pre-trained model for language representation, then fine-tuned it in supervised training stages using the available annotated data on the proposed classification job.

The content-based approach is the most popular family of approaches. Recent work on this family mainly uses deep learning techniques as we can notice from table 2.1.

2.1.2. Context-based approaches

The second family of approaches is the context-based method. Some authors also call this family of approaches the social context-based. People's social engagement analysis is emphasized by social context-based detection techniques, which involve the use of relevant social context elements reflecting users, posts, and network aspects of news consumption on social media. This family also has two sub-methods: the user-based and the propagation-based approaches [**49**].

Example of features for context-based approaches	
User-based	- Name - Job title - Political party affiliation - User behaviour on previous posts (e.g. user's actions, user's responses, user's emotions, user's browsing history, user's reactions, user's writing style, etc.).
Propagation-based	- The number of retweets - The number of comments - The number of likes - Friendship status - Relationships among the spreaders

Table 2.2. Example of features for context-based approaches

The user-based approach consists of using information about the author of the news to infer the validity of original news articles. The propagation-based approach focuses on the interrelationships of relevant social media postings for news credibility prediction [**49**]. Table 2.2 presents some features used to train a model based on the context. As reflected in table 2.3, just like the content-based approach, recent papers mainly propose deep learning-based solutions.

Author	**Used technique**	**Publish date**
Chen *et al.* [**15**]	Deep learning (LSTM)	2022
Jouyandeh *et al.* [**37**]	Deep learning (BERT)	2022
Xing *et al.* [**94**]	Deep learning (BERT and CNN)	2021
Sehgal *et al.* [**73**]	Machine learning (Naive Bayes)	2021
Dhall *et al.* [**23**]	Blockchain	2021
Huang *et al.* [**35**]	Deep learning (GRU, CNN)	2021
Li [**43**]	Machine learning (Logistic Regression)	2021
Gamallo [**26**]	Deep learning (BERT)	2021
Upadhyay *et al.* [**88**]	Deep learning (BERT)	2021
Megias *et al.* [**50**]	Deep learning (CCN)	2021
Raza [**62**]	Deep learning (Transformer)	2021
Ni *et al.* [**55**]	Deep learning (Attention mechanisms)	2021
Nagaraja *et al.* [**52**]	Machine learning (SVM)	2021
Lao *et al.* [**41**]	Deep learning (LSTM)	2021
Xue *et al.* [**96**]	Deep learning (MCNN)	2021

Table 2.3. A list of recent papers on Fake news detection based on the context

Sohan *et al.* [**83**] presented an approach that determines if an article is fake or real depending on the profile of the person who is spreading the information. Thus, the task is to detect fake news spreaders. To address this task, they used Decision Tree, Random Forest and Gradient Boosting classification algorithms.

To our knowledge, this work is the only paper in the past ten years that focuses on user profiles based on previous posts and reports good performance. It assigns a user to a category only based on his history. We had to go back to 2011 to find a paper from Castillo *et al.* [**12**] with significant results. However, again, only the source's history based on a regular average is used to determine its credibility.

On Twitter, news also propagates significantly differently than on other platforms. Meyers *et al.* [**51**] studied the potential of using propagation features to detect fake news on Twitter by using a non-URL-restricted data set. The study consisted of two main goals: Manually extract features from the propagation graphs to look at the probable significant discrepancies in how real and fake news spread on Twitter; Create two classifiers that have been trained on the propagation graphs. A classifier trained on the features manually retrieved and a geometric deep learning approach that is trained on the propagation graphs.

They manually extracted features such as the average number of followers, the average time between a tweet and a corresponding retweet, etc. To address the second goal, they trained the following classifiers: Random Forest, Decision Tree, Linear Discriminant Analysis, Bayes Neural Network, Logistic Regression, K-Nearest Neighbors, Quadratic Discriminant Analysis and Support Vector Machine. Then, the authors developed a neural network architecture for the geometric deep learning approach that was trained on the propagation graph. The nodes of this propagation graph are characterized by information like the number of retweets, the number of friends (following) of the user or the number of likes of the tweet/retweet.
This study reveals the following important distinctions in the dissemination of real and fake news: graphs of true news are larger in size, propagated by users with more followers and fewer followers, and stay on Twitter longer than fraudulent news.

Silva *et al.* [**81**] proposed Propagation2Vec which is a propagation-based fake news early detection technique that assigns variable levels of importance to nodes and cascades in propagation networks and reconstructs the knowledge of entire propagation networks based on their partial propagation networks at an early detection deadline.

Existing propagation-based detection approaches have two empirically validated research gaps, which Propagation2Vec aims to fill. First, most existing approaches fail to emphasize the informative nodes and cascades in propagation networks. To solve this, they offered a hierarchical attention technique for encoding propagation networks, which can assign various levels of priority to nodes/cascades. Second, most previous methods

rely on complete propagation networks, which are ineffective for the early identification of fake news. In response to this issue, they propose a technique to reconstruct the useful knowledge available in complete propagation networks using early propagation networks.

2.1.3. Hybrid-based approaches

The last approach family consists of a mix between the content-based family and the social context-based family. Mahid *et al.* [**49**] named it the hybrid-based family.

Author	Used technique	Publish date
Davoudi *et al.* [**18**]	Deep learning (LSTM)	2022
Lei [**42**]	Deep learning (LGAT)	2021
Liu *et al.* [**45**]	Deep learning (CNN and RNN)	2021
Rath *et al.* [**61**]	Machine learning (Graph neural network)	2021
Minha *et al.* [**39**]	Deep learning (Knowledge distillation KD)	2021
Sohan *et al.* [**83**]	Machine learning (Gradient boosting)	2021
Song *et al.* [**85**]	Machine learning (Graph neural network)	2021
Chalkiadakis *et al.* [**14**]	Machine learning (Random forest)	2021
Zhang *et al.* [**97**]	Deep learning (ResNet)	2021
Sheikhi [**75**]	Machine learning (WOA-Xgbtree)	2021
Silva *et al.* [**81**]	Machine learning (Graph - Propagation2Vec)	2021
Nguyen *et al.* [**54**]	Machine learning (Graph representation)	2021
Meyers *et al.* [**51**]	Machine learning (Random forest) Deep learning (Graph neural network GNN)	2021
Gangireddy *et al.* [**27**]	Machine learning (Unsupervised - Graph)	2021
Xie *et al.* [**93**]	Deep learning (Multi-head attention mechanism)	2021

Table 2.4. A list of recent papers on Fake news detection based on both the context and the content

In table 2.4, we can see that, in opposition to the previous families where deep learning techniques were the most used, authors of recent papers that propose a hybrid approach uses as much as deep learning techniques like machine learning techniques.

Li [**43**] presented an approach that uses the TF-IDF feature extraction technique and the stacking ensemble learning method based on weak classifiers. To increase the performance of the model, The author did not just analyze the content of the news, but also combined effective information such as publishers and topics. The weak classifiers used were Logistic Regression (LR), SGD Classifier (SGDC), Passive Aggressive Classifier (PAC), Ridge Classifier (RC) and Linear SVC (LSVC). A classification prediction result will be generated for each base model that we have just listed. The author used 5-fold cross-validation to train each base model, concatenate the classification prediction results after each base model training, and ultimately submit all the features to the final LightGBM model for training. LightGBM is an improvement of the Gradient Boosted Decision Tree (GBDT) algorithm.

NI *et al.* [**55**] focused on the source tweet and its propagation structure. They presented a novel neural network-based model, Multi-View Attention Networks (MVAN) to detect fake news and provide explanations on social media. The MVAN model contains text semantic attention as well as propagation structure attention, ensuring that the model can capture both source tweet content and propagation structure information and clues. Furthermore, the model's two attention methods can detect key clue words in fake news texts as well as suspicious users in the propagation structure.

Han *et al.* [**32**] presented a knowledge-based fake news detection that does not require any external knowledge graph. As illustrated in figure 2.6, the authors transformed the problem of detecting fake news into a subgraph classification task. Entities and relations are extracted from each news item to form a single knowledge graph, where each news item is represented by a subgraph. Then, to categorize each subgraph/news item, a graph neural network (subGNN) model is trained. They also report an improvement in the model's performance by combining extracted knowledge, textual content, propagation, and social context in a simple yet effective multi-modal technique.

Because of the inherent relationship between publishers, news, and social engagements during the process of news transmission on social media, a tri-relationship embedding framework TriFN was presented by Shu *et al.* [**80**]. Thus, this framework allows the detection of fake news by exploiting the social context. Tri-FN is composed of the five

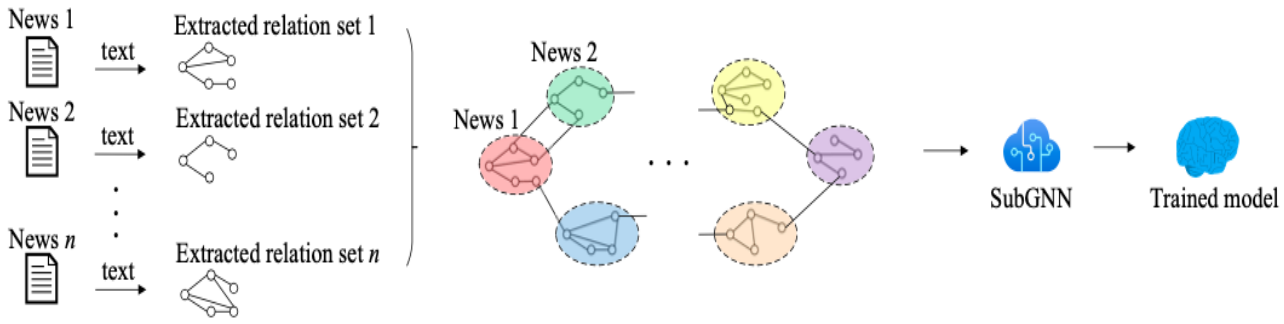


Fig. 2.6. An overview of the proposed knowledge-based fake news detection algorithm of Han *et al.* [**32**].

following components: a news contents embedding component, a user embedding component, a user-news interaction embedding component, a publisher-news relation embedding component, and a semi-supervised classification component.

The news contents embedding component describes how to map news from bag-of-words features to latent feature space; the user embedding component shows how to extract user latent features from user social relations; The feature representations of news pieces are learned by the user-news interaction embedding component based on their partial labels and user credibility; The feature representations of news pieces are regularised by the publisher partisan bias labels in the publisher-news relation embedding component; To predict unlabeled news items, the semi-supervised classification component learns a classification function. They tested the TriFN on two different datasets: BuzzFeed and PolitiFact.

Sahoo *et al.* [**70**], also propose a hybrid approach to automatically detect fake news for Facebook users based on the content of the news and the information about the user profile of the person who uploads the post. Thus, they've added the behaviour of multiple features associated with Facebook accounts. They have developed a chrome extension that uses machine learning and deep learning classifiers. The machine learning classifiers used are k-nearest neighbors algorithm KNN, SVM, Logistic Regression, Decision tree and Naïve Bayes. The only deep learning classifiers used are Long Short-Term Memory (LSTM).

Raza [**62**] suggested an approach that uses information from news articles and social contexts for fake news detection on political platforms. The proposed approach is built on a Transformer architecture that can learn meaningful representations from fake news data and predicts whether a news item is fake or not. The author modified the structure of pre-trained Bidirectional Encoder Representations from Transformers (BERT) to add side information such as other users' reactions to the news. The same procedure can be

used on other transformers like BART, XLNet and RoBERTa. The author named this transformer-based model Faker and used the NELA-GT-2020 dataset. The possible labels for this dataset are reliable, mixed and unreliable. Raza [**62**] transformed this dataset into a binary classification problem with two possible labels fake or real.

Song *et al.* [**85**] presented a temporal propagation-based fake news detection framework named Temporally Evolving Graph Neural Network for Fake News Detection (TGNF). This framework can fuse structure, content semantics, and temporal information. By modelling the node's temporal interaction events, it can model real-world news temporal evolution patterns as a graph growing in the context of continuous-time dynamic diffusion networks. Three real-world datasets were used during this study: Weibo, FakeNewsNet and Twitter datasets. They concluded that modelling and adding information on the temporal propagation of online social media news can aid in the detection of fake news.

2.1.4. Explainable Fake News Detection

This section will focus on the explainability of fake news detection. Recently, a large number of papers in the industry are proposing modules explaining their systems. Except for Silva *et al.*'s paper [**81**], none of the previously listed papers provide an explanation or illustration to users to help them understand why a piece of news is flagged as false or true. Only a few recent papers have come up with an explainability module.

Shu *et al.* [**77**] presented dEFEND, an explainable fake news detection system that uses an attention neural network. The proposed system provides an explanation based on news content and user comment as shown in figure 2.7. A co-attention mechanism is used to jointly capture the intrinsic explainability of the posts and the comments.
Lu *et al.* [**46**] proposed a neural network-based model named Graph-aware Co-Attention Networks (GCAN) which detects if the input data is fake and produce an explanation by highlighting the evidence of questionable retweeters and the words they mention.

Przybyła *et al.* [**60**] introduce a system that computes the credibility score based on the stylometry of news and can explain the logic behind the used automated algorithms and increase the reader's trust in their credibility evaluation. The system used two machine learning approaches: one is a linear algorithm trained on stylometric data, and the other is a recurrent neural network.

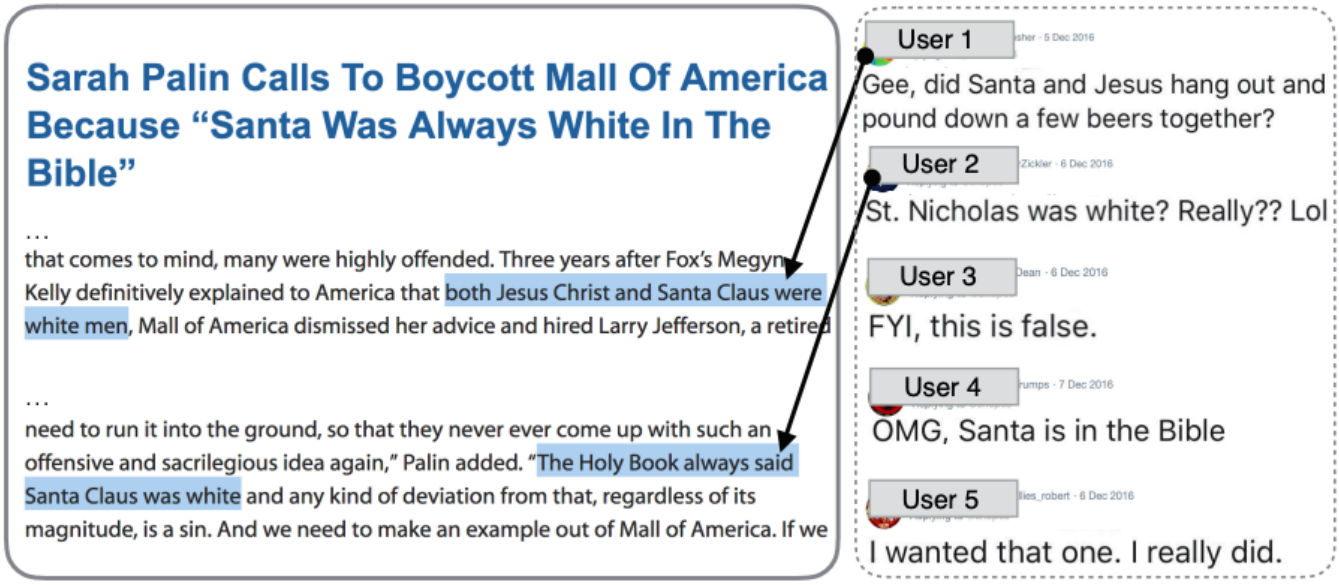


Fig. 2.7. A piece of fake news on PolitiFact, and the user comments on social media. Some explainable comments are directly related to the sentences in news contents [**77**]

Reis *et al.* [**65**] presented an explainable machine learning for fake news detection that provide an explanation of the aspects that influence model decisions, enhancing civic reasoning by enhancing our capacity to analyze digital content and reach warranted conclusions. The authors used a unified framework for interpreting predictions named SHAP (SHapley Additive exPlanations) [**47**].

2.2. Dataset

Fake news detection problem is increasingly becoming the subject of huge number work in the industry. Although there are several approaches as we saw in the previous chapter, one of the main issues that searchers are facing is the lack of comprehensive and community-driven fake news databases. Not only are existing datasets sparse, but they also lack a variety of aspects commonly needed in research, such as news content, social context, and spatial data [**74, 79**]. Shu *et al.* [**79**] presented a data repository named FakeNewsNet to mitigate the lack of quality data sets. FakeNewsNet is the dataset used in our works. In the following subsections, we are going to describe the used dataset and the preprocessing of the data.

2.2.1. Dataset description

The main difference between FakeNewsNet and other datasets made for the fake news detection task is the diversity of the available features. To our knowledge, FakeNewsNet is

the only dataset that contains news content (text and image), social context (information about users, the network, etc.) and spatiotemporal information (locations, timestamps, etc.).

FakeNewsNet contains data collected from PolitiFact and GossipCop two well-known fact-checkers websites. Thus, this dataset includes news articles URL with labelled tweets related to this those news articles. On top of that, as we said before, FakeNewsNet has more features, as shown in Table 2.5, which shows the statistical data collected from PolitiFact.

	Category	**Features**	**PolitiFact**	
			Fake	**Real**
News Content	Linguistic	# News articles	432	624
		# News articles with text	420	528
	Visual	# News articles with images	336	447
Social Context	User	# Users posting tweets	95,553	249,887
		# Users involved in likes	113,473	401,363
		# Users involved in retweets	106,195	346,459
		# Users involved in replies	40,585	18,6675
	Post	# Tweets posting news	164,892	399,237
	Response	# Tweets with replies	11,975	41,852
		# Tweets with likes	31,692	93,839
		# Tweets with retweets	23,489	67,035
	Network	# Followers	405,509,460	1,012,218,640
		# Followees	449,463,557	1,071,492,603
		Average # followers	1299.98	982.67
		Average # followees	1440.89	1040.21
Spatiotemporal Information	Spatial	# User profiles with locations	217,379	719,331
		# Tweets with locations	3,337	12,692
	Temporal	# Timestamps for news pieces	296	167
		# Timestamps for response	171,301	669,641

Table 2.5. Statistics of the FakeNewsNet repository

PolitiFact is a fact-checkers website specializing in politics. Journalists and domain experts review political posts (tweets) and give one of the following labels: pants on fire, false, barely true, half true, mostly true, and true. On another hand, Gossip Cop is a fact-checkers website specializing in entertaining stories. GossipCop assigns a rating to a news story on a scale of 0 to 10 to indicate the degree to which it is fake or true. The higher the score, the more the news is true.

Our work helps this dataset to have more diversity with more social context features. First, our system extracts named entities from input text. The Twitter usernames of the most often referenced named entities in the used dataset were manually added to a database. Once a named entity is extracted and our database contains his Twitter username, DS-Fake

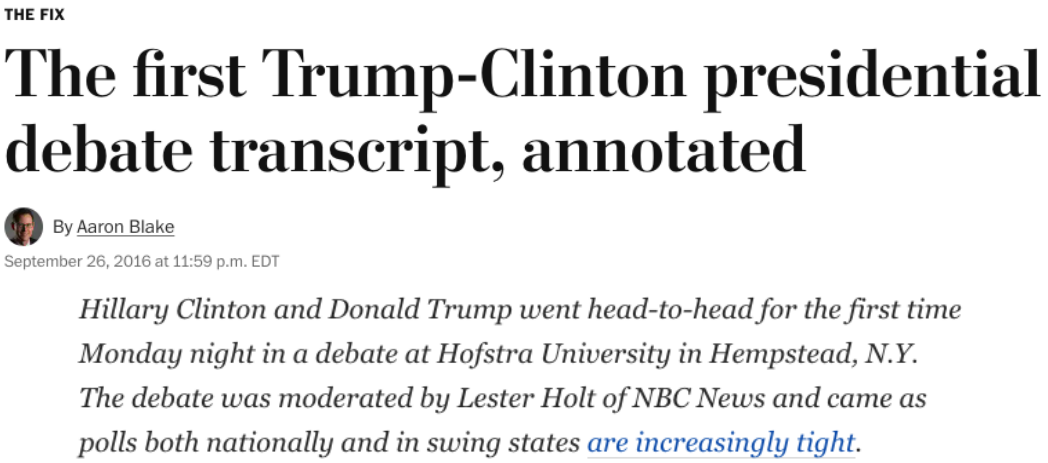
receives a flow of data from this Twitter account.

Secondly, each source whose at least one post has been given to the system as an input receives a credibility score that we named the legitimacy score. The next section presents how this score is computed. Thus, posts from named entities mentioned in the input and the legitimacy score enrich the FakeNewsNet dataset with more social context features.

2.2.2. Dataset processing

As we can see in table 2.5, FakeNewsNet contains a considerable number of data. Due to the time complexity of training and fine-tuning our models, we decided to only keep features that we are going to use for our experience (as presented in sections 4.2.2 and 4.2.3). We only used the entire dataset to compare the performance of DS-Fake with other approaches on this dataset (as presented in section 4.2.4).

We decided to put our focus on data from PolitiFact. Moreover, among the data, we choose to only focus on articles and tweets published or posted during the year 2016 or directly related to this specific year. By directly related to the year 2016, we mean, for example, a tweet posted at the beginning of the year 2017 or the end of the year 2015 but related to articles published in 2016.


THE FIX

The first Trump-Clinton presidential debate transcript, annotated

By Aaron Blake

September 26, 2016 at 11:59 p.m. EDT

Hillary Clinton and Donald Trump went head-to-head for the first time Monday night in a debate at Hofstra University in Hempstead, N.Y. The debate was moderated by Lester Holt of NBC News and came as polls both nationally and in swing states are increasingly tight.

Fig. 2.8. A news article checked by PolitiFact

We specifically chose the year 2016 because, during the US presidential election campaign of that year, the propagation of false news on social media became a national concern in the United States and the rest of the world [**30**]. One of the most talked-about aspects of the 2016 US Presidential Election was the proliferation of content created by fake news

Approch	Source	Features
Content-based	News article	The text content of the article
	Post (tweet)	The text content of the tweet
Context-based	News article	Author's name
		Published date and time
	Post (tweet)	Author's name
		Published date and time

Table 2.6. Features of the transformed dataset

publishers [**9, 30**]. Fake news is also said to have influenced the 2016 US presidential election [**9, 79**].

Thus, we kept all the news articles, including information about the author as well as the publish date and time. Figure 2.8 shows an example of a news article (title and first paragraph) of this dataset checked by PolitiFact and labelled as True.

We also kept all tweets, including information about the author (user), the label, as well as the publication date and time. These tweets are the input of DS-Fake. Figures 2.9 and 2.10 present tweets extracted from this dataset. Figure 2.9 is a tweet labelled as True related to the article shown in figure 2.8. Figure 2.10 is a tweet labelled as False related to the same article. Table 2.6 shows the result of the feature transformation.

Fig. 2.9. An example of a tweet labelled as True extracted from the dataset.

Fig. 2.10. An example of a tweet labelled as False extracted from the dataset.

We mentioned earlier that DS-Fake receives a flow of data (tweets) of mentioned named entities extracted from the input. For example, in the inputs presented in figures 2.9 and 2.10, the named entity 'Hillary Clinton' can be extracted. Thus, the system retrieves tweets from her official Twitter account. Figure 2.11 shows an example of a tweet related to these inputs and the article in figure 2.8.

Fig. 2.11. An example of a tweet labelled as False extracted from the dataset.

The preprocessing process is as follows: all the texts are transformed to lower case, then stop words, punctuation signs, emojis and symbols are removed. After that, the data is transformed into tokens. The last step was to stem our dataset.

2.3. Conclusion

This chapter presented different approaches related to the fake news detection problem and shows up the main contribution of our proposed system. Most of the recent works use machine learning or deep learning, but none proposed an approach that receives a flow of data. Our work is the first that uses a data stream approach for the fake news detection problem. Our system receives data streams from fact-checkers websites and Twitter accounts. The received data from Twitter add features to the used dataset. The last section

of this chapter presented the dataset. Furthermore, it showed how this work improves the diversity of this dataset with new social context features thanks to the legitimacy score based on the history of all the input posts and the receive flow data from mentioned named entities within the input.

Additionally, this chapter showed that only a few papers provide an explanation. Among these, most use the co-attention mechanism or machine learning method to highlight words within the input text that contributed the most to obtained result. Our explanation module highlights important words in the text of the trusted articles and the legitimate posts. Moreover, more information is provided, such as the source, the published date and time, and the link to the retrieved trusted articles from fact-checkers websites and legitimate posts from mentioned named entities.

Papers that used user-based methodologies were also covered in this chapter. Only a few works use the post history of a source to automatically determine his credibility. The majority uses features such as the writing style, the number of followers, the job title, etc. Our approach is the first to use the Bayesian average metric. This average includes the post history of all sources to determine the legitimacy score of each source. The Bayesian average helps to reduce the impact of previous tweets according to the number of tweets in history.

In this chapter, a diversity of approaches was introduced. However, none uses the combination of the text similarity task and natural language inference to mitigate this problem.

The next chapter presents DS-FAKE, our proposed system.

Chapter 3

The proposed DS-FAKE system

This chapter discusses the general architecture of our proposed DS-Fake system. This chapter highlights the novelty of our proposed system, which is the first based on data stream mining. The previous chapter presents papers that explain the obtained results, this chapter shows how our explanation module differs in terms of the quantity of return information, words that are highlighted and so on. This chapter explains in detail how the legitimacy score is computed based on the history of all sources in opposition to the way most authors determine the credibility of a source based on features such as the localization, the username or the job title. Furthermore, it will show how we combined the text-similarity and the natural language inference task for the fake news detection problem.

3.1. DS-Fake

In this section, we are going to present our proposed system named DS-Fake. We will start with a general overview of how the system works. After that, we will introduce the general architecture of DS-Fake. The following subsection will present the implementation of this system.

3.1.1. General Overview

The main goal of our work is to have a near real-time system that detects fake news and reduces future propagations of fake news by educating the users of this system. To achieve this goal, we built a system named DS-Fake (Data Stream - Fake). This system, as its name suggests, is a system based on a continuous data collection, a continuous flow of data.

As we mentioned in the previous chapter, we distinguish three families of automatic approaches (see figure 2.1) [**49**]. Among three families, none of the presented works is based on a flow of data received continuously for a certain period. The data stream mining approach for the fake news detection problem is a new type of model to explore. To our

knowledge, this work is one of the first works that address this aspect. DS-Fake belongs to the family of hybrid approaches since it uses the content (the input and the streamed text) and the context (the legitimacy score).

Even though this model can be extended to any platform, due mainly to the available data, we choose to only focus on Twitter. Thus, the input of this system is a post from Twitter. DS-Fake exploits the text of the tweet, information about the user that posted it and the posted date and time.

The principal idea behind this system is to stream data from trusted and/or legitimate sources related to the input post that we want to determine reliability. DS-fake returns a percentage of confidence that an input post is reliable. The higher percentage, the more we can trust the text content of the tweet. The percentage that is returned is obtained from a combination of scores coming from three different modules of the system: DS-Fact, DS-Source and DS-Entity. We will see the implementation of those modules in the following subsection.

In reality, this system returns three percentages at three different time steps. The first percentage is returned after a request is received. Even though a percentage is returned, DS-Fake will keep receiving stream data. After a certain amount of time, another percentage will be computed. This new percentage will reflect the computation from both, old data and new data that were received after the first result. Afterwards, the system will keep retrieving new data related to the input. The third and last percentage will be computed and returned after a while following the same idea as at the second timestep.

Once the system receives a request, all three modules are executed simultaneously in order to compute a percentage. At each timestep, to compute the percentage, the three first modules work as follows:

- The first module is DS-Fact. It is in charge of computing a score based on trusted sources such as fact-checking websites. Our dataset contains URLs of news articles posted or checked by PolitiFact. Therefore, our system only takes into account news from PolitiFact.com. The first step of this module is to find articles that are closely related to the input post between all the past and coming articles from PolitiFact. With the closest articles, this module computes a score based on the inference relationship with the input.

- The second module, DS-source, is based on the context of the input post. Thus, it returns the legitimacy score of the user that posted the tweet. Most credibility scores in the recent work are determined based on features such as the user's emotions on previous posts, his job, his writing style, etc. There are almost no authors that used the post history. When the post history is used, the credibility score assigned to a source only takes into account previous tweets of this source. While our credibility score, named the legitimacy score, considers the history of the percentages of each post of all the sources to determine the score of each source by using the Bayesian average. Thus, DS-source keeps previous percentages of tweets posted by each user. After each complete execution of the system after a request, the score of all users is updated. The more a user posts reliable tweets, the higher their legitimacy score.

- DS-Entity is the third module. It works slightly the same as the first module DS-Fact. The main difference is the source of the external needed data. For this module, contrarily to the first module that receives a data stream from a fact-checkers website, DS-Entity receives a data stream from legitimate sources. By legitimate sources, we mean, the named entities mentioned in the input post. In consequence, the first step of this module is to extract all the named entities from the input tweet. The second step is to start retrieving posts from those named entities. For the following step, the idea is the same as the first module. DS-Entity find close tweet posts from the account of the extracted named entities. With the closest tweets, it computes a score based on the inference relationship with the input.

This encapsulation which consists of finding the closest text (articles or text) and then determining the inference from the closest retrieved texts in terms of text similarity is the first time that this type of combination is used for this problem.

DS-Fake also has a module named DS-Explain. The first function is to compute the percentage of confidence based on the score of the three previous modules. This module is also charged to provide an explanation to the users of the system. With the explanation function, the user can understand the percentage that his input receives. This module returns the percentage and a simple explanation that may help users adopt a certain behaviour. The idea is to make users start checking if a post is reliable themselves before sharing the post. We will see in the next subsection how our explanation module differs from other works.

The last part of this section presents a scenario of DS-Fake. The input is the post shown in figure 2.9.

DS-Fake input
Text content: Clinton: Trump called women dogs: At the first presidential debate at Hofstra University, Hillary… #Skibabs Source: @Skikabs Date & Time: 2016-09-26 11:50 PM

DS-Fake

Timestep 0 | t=0

Module DS-Fact:

Input at t=0 →This module receives as an input the DS-Fake input tweet' content.

Steps at t=0 →

1. Once the input is received, the module starts receiving a data stream from a trusted source (it receives the content of articles). The goal is to find reliable articles related to the input tweet. Thus, the first step is to scrape PolitiFact.com to find articles;
2. The second step is to find among the retrieved articles, articles that are similar to the topic of the input by using a text similarity technique;
3. The last step is to compute a score based on the number of closest articles that infer the input tweet.

Output at t=0 → This module returns the score computed at step 3 of this module.

At this timestep, no relevant articles are found. Thus, this module returns a score of 50%. A score of 50% means that the input has a 50% of chance being True and a 50% of chance False. A first score is returned. However, DS-Fact continues to scrape PolitiFact.com until a second score will be computed at the next timestep.

Module DS-Source:

Input at t=0 → This module receives as an input the Twitter username of the author of the tweet given to DS-Fake.

Steps at t=0 →

1. The first and only step of this module at this timestep is to return the corresponding legitimacy score of the input source (username).

Output at t=0 → This module returns the legitimacy score.

The module doesn't recognize the user "@Skikabs". It's the first time that the system receives input from this user. Thus, a score of 50% is returned. A score of 50% means that the user has a 50% of chance being a trusted source and a 50% of chance not being a trusted source.

Module DS-Entity:

Input at t=0 → This module receives as an input the DS-Fake input tweet' content.

Steps at t=0 →

1. The module extracts all the named entities mentioned in the input tweet;
2. The second step is to start receiving data streams from the official Twitter account of the extracted named entities (it receives the content of tweets);
3. The next step is to find between the retrieved tweets, tweets that are similar to the topic of the input;
4. The last step is to compute a score based on the number of closest tweets that infer the input tweet.

Output at t=0 → This module returns the score computed at step 4 of this module.

The module finds the following named entity: Trump, Hofstra University, and Hillary. DS-Fake has a database of the official certified Twitter accounts of the most mentioned name entity of our dataset. DS-Entity starts to receive data streams from the official certified account of Donald Trump (@realDonaldTrump) and Hillary Clinton (@HillaryClinton). The module finds similar tweets from named entities that infer the input tweet. Here is an example of a tweet from @HillaryClinton published on September 26, 2016, at 10:01 PM.

Based on all the retrieved tweets that infer the input, a score of 84% is returned. A score of 85% means that the input text has an 84% of chance being True. A first score is returned by this module. However, DS-Entity continues to receive tweets from extracted named entities.

Module DS-Explain:

Input at t=0 → This module receives as an input the score computed in the three previous modules (DS-Fact: 50%; DS-Source:50%; and DS-Entity:84%).

Steps at t=0 →

1. The first and only step of this module at this timestep is to compute the global percentage based on the three input scores.

Output at t=0 → The global percentage. The first percentage that is returned by DS-Fake.

At this timestep, the global percentage is 71%.

The global percentage returned by DS-Fake is: 71%

Between timesteps 0 and 1, there is a sleeping time during which no calculation is performed. However, DS-Fake modules (DS-Fact and DS-Entity) continue to receive data stream.

Timestep 1 | t=1

Module DS-Fact:

Input at t=1 → This module receives as an input the DS-Fake input tweet' content.

Steps at t=1 →

1. The first step is to find similar articles among those retrieved from PolitiFact.com from the beginning of the timestep 0 until now;
2. The last step is to compute a score based on the number of closest articles that infer the input tweet.

Output at t=1 → This module returns the score computed at step 2 of this module.

At this timestep, DS-Fact finds close articles that infer the input tweet. Here is an example of a corresponding article (the title and a relevant part of the body of the article published on September 26, 2016, at 11:59 PM).

The first Trump-Clinton presidential debate transcript, annotated

By Aaron Blake

September 26, 2016 at 11:59 p.m. EDT

> CLINTON: You know, he tried to switch from looks to stamina. But this is a man who has called women pigs, slobs and dogs, and someone who has said pregnancy is an inconvenience to employers, who has said...
>
> TRUMP: I never said that.

Based on all the retrieved articles that infer the input, a score of 90% is returned. A second score is returned by this module. However, DS-Fact continues to receive a data stream from the fact-checker website.

Module DS-Source:

Input at t=1 → This module receives as an input the Twitter username of the author of the tweet given to DS-Fake.

Steps at t=1 →

The first and only step of this module at this timestep is to return the corresponding legitimacy score of the input source (username).

Output at t=1 → This module returns the legitimacy score.

The module still doesn't recognize the user "@Skikabs". Thus, a score of 50% is returned.

Module DS-Entity:

Input at t=1 → This module receives as an input the DS-Fake input tweet' content.

Steps at t=1 →

1. The module has already extracted all the named entities mentioned during the last timestep. Here, the first step is to find similar tweets

among those retrieved from named entities' official accounts from the beginning of the timestep 0 until now;
2. The last step is to compute a score based on the number of closest tweets that infer the input tweet.

Output at t=1 → This module returns the score computed at step 2 of this module.

DS-Entity finds more similar tweets from named entities that infer the input tweet. Based on all the retrieved tweets that infer the input, a score of 89% is returned. A second score is returned by this module. However, DS-Entity continues to receive tweets from extracted named entities.

Module DS-Explain:

Input at t=1 → This module receives as an input the score computed in the three previous modules (DS-Fact: 90%; DS-Source:50%; and DS-Entity:89%).

Steps at t=1 →

1. The first and only step of this module at this timestep is to compute the global percentage based on the three input scores.

Output at t=1 → The global percentage. The second percentage that is returned by DS-Fake.

At this timestep, the global percentage is 89%.

The global percentage returned by DS-Fake is: 89%

Between timesteps 1 and 2, there is a sleeping time during which no calculation is performed. However, DS-Fake modules (DS-Fact and DS-Entity) continue to receive data stream.

Timestep 2 | t=2

Module DS-Fact:

Input at t=2 → This module receives as an input the DS-Fake input tweet' content.

Steps at t=2 →

1. The first step is to find similar articles among those retrieved from PolitiFact.com from the beginning of the timestep 0 until now;
2. The second step is to compute a score based on the number of closest articles that infer the input tweet;
3. The last step is to stop receiving data flow from PolitiFact.com.

Output at t=2 → This module returns the score computed at step 2 of this module.

At this timestep, DS-Fact finds more close articles that infer the input tweet. Based on all the retrieved articles that infer the input, a score of 96% is returned.

Module DS-Source:

Input at t=2 → This module receives as an input the Twitter username of the author of the tweet given to DS-Fake.

Steps at t=2 →

1. The first of this module at this timestep is to return the corresponding legitimacy score of the input source (username);
2. The last step is to update the legitimacy score of the source after the final global percentage is computed.

Output at t=2 → This module returns the legitimacy score.

The module still doesn't recognize the user "@Skikabs". Again, a score of 50% is returned.

Module DS-Entity:

Input at t=2 → This module receives as an input the DS-Fake input tweet' content.

Steps at t=2 →

1. The module has already extracted all the named entities mentioned during the first timestep. Here, the first step is to find similar tweets among those retrieved from named entities' official accounts from the beginning of the timestep 0 until now;
2. The second step is to compute a score based on the number of closest tweets that infer the input tweet;
3. The last step is to stop receiving data flow from named entities.

Output at t=2 → This module returns the score computed at step 2 of this module.

DS-Entity finds more similar tweets from named entities that infer the input tweet. Based on all the retrieved tweets that infer the input, a score of 93% is returned.

Module DS-Explain:

Input at t=2 → This module receives as an input the score computed in the three previous modules (DS-Fact: 96%; DS-Source:50%; and DS-Entity:93%).

Steps at t=2 →

1. The first of this module at this timestep is to compute the global percentage based on the three input scores;
2. The second step is to produce an explanation for the user that sent the request. The module highlights significant words that lead to the returned final global percentage.

Output at t=2 → The final global percentage. The last percentage is returned by DS-Fake and the explanation generated at step 2.

At this timestep, a global percentage is 96%.

The global percentage returned by DS-Fake is: 96%

DS-Fake final output
Text content: Clinton: Trump called women dogs: At the first presidential debate at Hofstra University, Hillary… #Skibabs Source: T@Skikabs Date & Time: 2016-09-26 11:50 PM Final global percentage: 96% Explanation → DS-Fake find the following article from a trusted fact-checkers website (PolitiFact): Title: "The first Trump-Clinton presidential debate transcript, annoted" Body: "Hillary Clinton and Donald Trump went head-to-head for the first time Monday night in a debate at Hofstra University in Hempstead, N.Y. …But this is a man who has called women pigs, slobs and dogs, and someone who has said pregnancy is an inconvenience to employers, who has said..." Date & Time: 2016-09-26 11:59 PM Click here for the link to the article DS-Fake find the following tweet from a named entity mentioned in the input (@HillaryClinton): "Trump just criticized me for preparing for this debate. You know what else I prepared for? Being president." Date & Time: 2016-09-26 10:01 PM Click here for the link to the Twitter post

The next subsection will present in further detail the general architecture and the implementation of those modules.

3.1.2. General Architecture

This subsection will introduce the general architecture of our proposed model named DS-Fake. We will also present the implementation of each module of this system

DS-Fake has four principal modules. As we can see in figure 3.1, those modules are DS-Fact, DS-Source, DS-Entity and DS-Explain. In the previous subsection, we have seen the purpose of each of these modules. Thus, we are going to see how these modules work together.

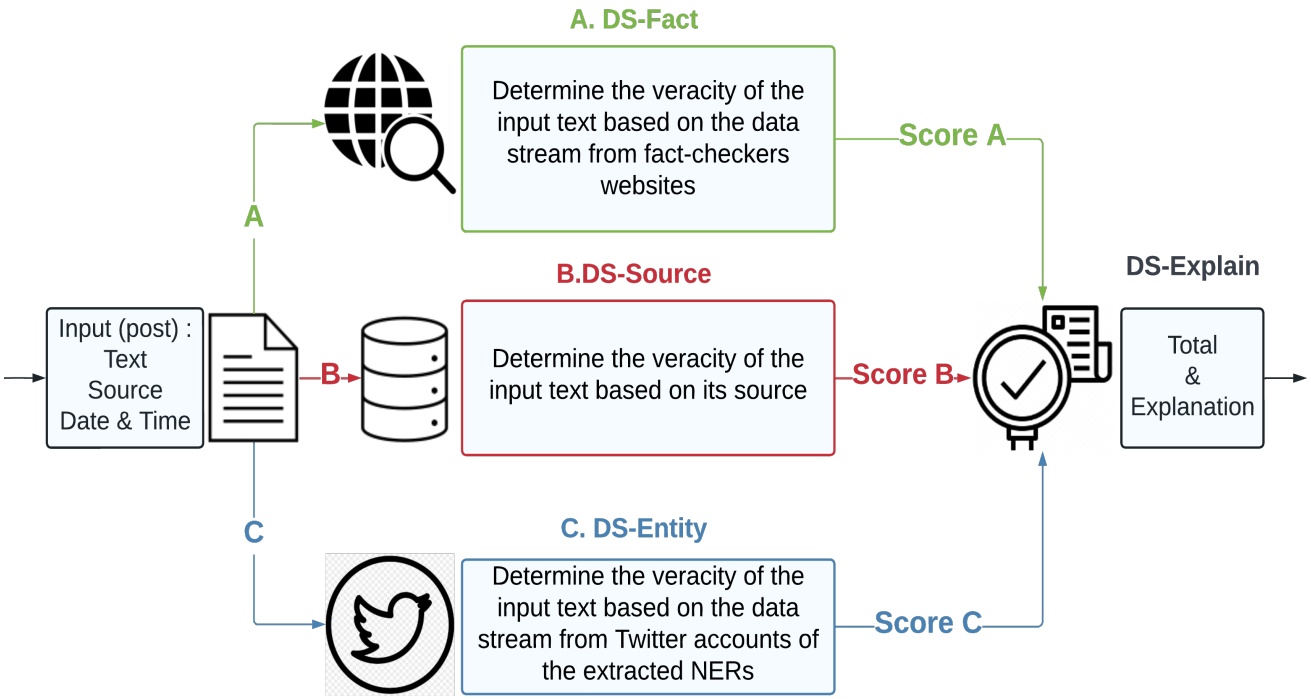


Fig. 3.1. DS-Fake Architecture

We mentioned earlier that DS-Fake receives a data stream from fact-checkers websites even when there is no request. Therefore, DS-Fake has a queue with recent articles that have been reviewed or published by a fact-checker. So, when the system receives a request, the DS-Fact module can skip the first step, which consists of scraping the fact-checker website. This behaviour is desired because it tends to reduce the system's execution time.

The pseudocode shown in algorithm 3.1.1 presents the execution of DS-Fake once a request is received. The input of DS-Fake is a post on Twitter. The first step is to do some extraction and preprocessing. The system extracts the published date and the published time of the input tweet. Regarding the preprocessing, we presented it in section 2.2.2.

The second step is represented by the for loop in algorithm 3.1.1. This loop will be executed three times. Each of these executions represents a timestep. DS-Fake returns a percentage at three distinct timesteps. At the first execution, the modules DS-Fact, DS-Source and DS-Entity are launched simultaneously. Those modules are going to compute a score as described in the previous subsection.

Algorithm 3.1.1. This pseudocode shows the execution of DS-Fake once a request is received.

---

preprocessing()
$t = 0$
From 0 **to** 2 **:**
    Do in parallel :
        DS_Fact()
        DS_Source()
        DS_Entity()
    End do in parallel
    sleep(t)
    DS_Explain(t) *#compute the returned percentage and generate the explanation*
    $t = t + 1$
End for
update()
Return (percentage, explanation)

The next line is the sleep() function. This function doesn't stop the data stream but freezes the execution of the loop. We added this function in order to separate the computation of the percentage between each timestep. This function receives as a parameter the variable t. This variable specifies to this function at which timesteps of the execution of the system we are. Depending on the current timestep, the sleep() function stops the execution for a certain amount of time. During the freezing time, DS-Fact and DS-Entity continue to receive data streams from fact-checkers websites and accounts of named entities extracted from the input text. In other words, this function controls the execution time of

each module which is bounded by the duration of each timestep. The modules are launched simultaneously and after X minutes (at the end of each timestep), each module stops mining data and a score is computed and returned. If a module could not get a result, then a neutral score of 50% is returned. DS-Fact and DS-Entity continue to receive new data until the final score is computed.

Once all the scores are computed, the module DS-Explain is activated. This module receives as a parameter the variable t. We know that DS-Explain has two main functions: computing the returned percentage and generating the explanation. However, we saw in the scenario in the last subsection that the explanation is only generated and provided at the last timesteps. Consequently, we need to specify to this module at which timesteps of the execution of the system we are. We can do it by passing the variable t, which represents the current timestep. The last instruction of this loop is simply the incrementation of the variable t.

The last function is the update() function. This function will start by adding the Twitter account user that published the input tweet to the database of users if is not yet included. This database contains the history and the legitimacy score of all users whose at least one post was given as input to the system. Next, the function will update the legitimacy score of all users of the DS-Fake system based on the result of the current request.

The idea behind the legitimacy score follows the same logic as the five starts system of e-commerce websites [**71**]. Thus, the technique used for the legitimacy score is the Bayesian average [**1, 71**]. It returns a value between 0 and 100. The higher the legitimacy score is, the more post from this user/source can be trusted.

We especially chose this technique because it takes into account the number of previous posts received from users. The Bayesian average ensures that users with lower numbers of received posts (below a threshold) have less weight in the ranking. In order words, it adjusts the legitimacy score of users whose number of treated posts is under the threshold while it applies a slight change to the average rating of users whose number of posts is above the threshold.

For example, user A has one post in the history and this post received a final percentage of 100%. User B has ten posts in the history and all of them received a final percentage of 100%. If we compute the mean of these users, there will both have a legitimacy score of 5. However, user B tends to be more worthy of trust than user A since the system treated more posts from him. With the Bayesian average, the legitimacy score of user B

will be higher than user A.

The formula to compute the legitimacy score of a user/source is as follows:

$$LS = \frac{M * N + D * E}{N + E}$$

Where: LS is the legitimacy score, M is the mean of the percentage of all the received posts from this user/source, N is the number of posts received from this user/source, D is the mean of all percentages across the whole database and E is the minimum number of received post to be listed. We set E to 1.

In the following subsections, we are going to introduce in more detail each of the modules of our system.

3.1.2.1. DS-Fact. The first module that we are going to present is DS-Fact. As we saw in section 3.1.1, this module is in charge of receiving a data stream from fact-checkers websites and returning a score based on that. Figure 3.2 shows how this module is organized.

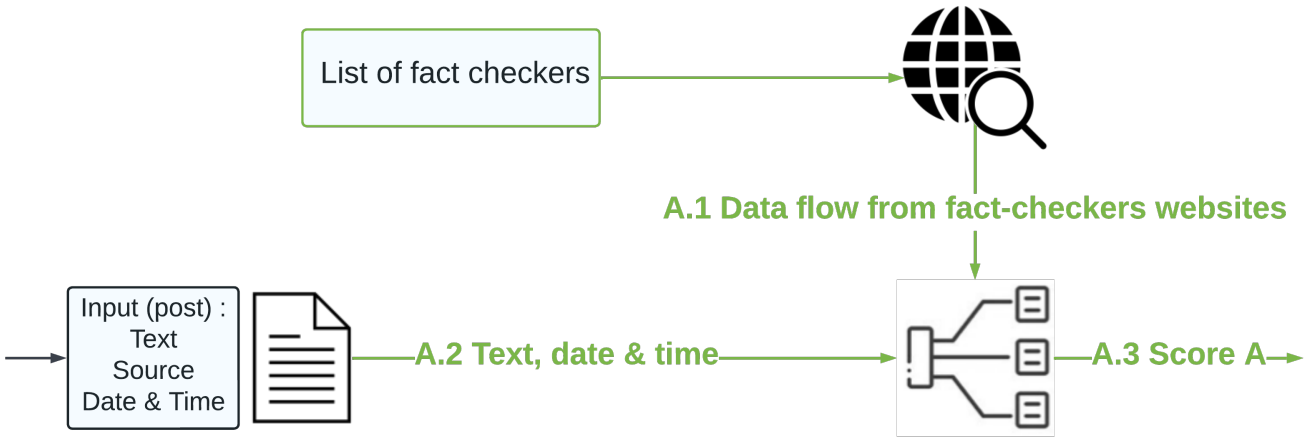


Fig. 3.2. DS-Fact Module

DS-Fact has a list of all fact-checker websites from which the system receives a data stream of trusted articles. Due to the limitation of our dataset, this list only contains PolitiFact.com.

So, when there is a new request. DS-fact receives the input text, the published date and the published time. With the help of these pieces of information, the first task of this module is to find the closest articles related to the input text. The date and time help the

module to focus its research on a specific period. The module starts receiving data streams from a certain period before the input publication date and time until the last percentage is computed. This task is equivalent to the text similarity task.

Between the huge variety of approaches that exist for the text-similarity task, we arbitrarily chose to use a pre-trained BERT model. Devlin *et al.*[**22**] introduced BERT as a pre-trained model that can be fine-tuned for different tasks. BERT, as presented by its authors, can also be fine-tuned for the text similarity task. Nevertheless, fine-tuning a BERT model for finding the similarity between two sentences will result in a large computational overhead because it necessitates feeding both sentences into the network. For example, if we want to find the most comparable pair in a batch of 10,000 sentences, it requires roughly 50 million inference computations. It can take approximately 65 hours.[**64**]

In order to fine-tune BERT in an efficient way for the text similarity task, we decided to follow the implementation published by Reimers *et al.* [**64**]. The authors presented a modification of BERT named Sentence-BERT (SBERT). SBERT is a pre-trained BERT network that uses siamese and triplet network architectures to generate semantically relevant sentence embeddings. With the same accuracy as the original BERT, SBERT reduces the time it takes to find the most similar pair from 65 hours to around 5 seconds. The term 'siamese network architecture' was first presented by Bromley *et al.* [**8**]. It is a network that contains two or more identical sub-networks that are connected at their outputs.

Figure 3.3 [**64**] shows the used architecture of SBERT for the text similarity task where u and v are the sentence embeddings of the input sentences A and B. These sentences A and B are the texts that we want to know how similar there are. cosine-sim stand for cosine similarity. We fine-tuned SBERT using the STS benchmark (Semantic Textual Similarity) [**13**], a benchmark on sentence similarity. Table 3.1 presents the used hyper-parameters for the training phase.

Hyper-Parameter	**Value**
Epochs number	4
Optimizer	Adam optimizer
Learning rate	2e-5
Weight decay	0.01
Regression loss	Mean squared-error loss
Batch-size	16
Random seeds	10

Table 3.1. SBERT training hyper-parameters for the text similarity task

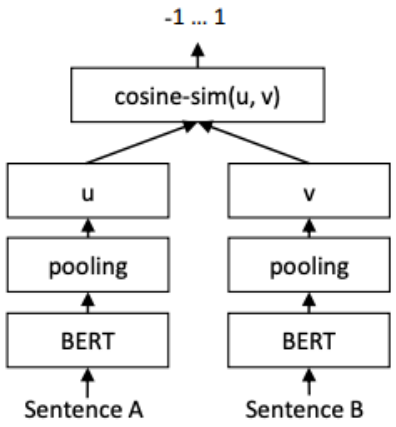


Fig. 3.3. SBERT architecture for the text similarity task

The output score of the text similarity task is between -1 and 1. We normalize the range from -1 to 1 into 0 to 1. We consider that an article is close to the input tweet if the normalized score is above or equal to the threshold of 0.8. With all the articles that reach the threshold, the next step is to determine if those closest articles can infer the input text. This amounts to doing the natural language inference task (NLI) task.

Storks *et al.* [**87**] summarised the different approaches for this task into 4 groups: symbolic approaches, statistical approaches, neural approaches, and knowledge-based approaches. For our system, we chose a neural approach since it's the most recent group that does not require external knowledge and this type of approach outperforms other approaches in almost every scoreboard for this task. Once more, we used a pre-trained SBERT model [**64**]. The NLI task was among the tasks presented by Devlin *et al.* [**22**], the authors that introduced BERT, that can be fine-tuned from it.

Figure 3.4 [**64**] shows the architecture of SBERT for the NLI task where u and v are the sentence embeddings of the input sentences A and B. Here, sentence A is the premise and sentence B is the hypothesis. |u-v| is the absolute element-wise difference of u and v. We combine u, v, and |u-v| to make a single long vector. This long vector captures information from the premise as well as the hypothesis. A softmax classifier is then used to predict our three classes using this vector (entailment, neutral and contradiction). For this task, SBERT was fine-tuned using the SNLI dataset [**7**]. Table 3.2 presents the used hyper-parameters for the training phase.

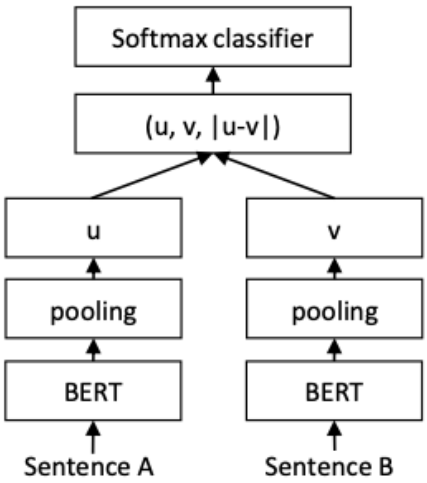


Fig. 3.4. SBERT architecture for the NLI task

Hyper-Parameter	Value
Epochs number	5
Optimizer	Adam optimizer
Learning rate	1e-5
Weight decay	0.01
Regression loss	Sparse categorical cross-entropy loss
Batch-size	128

Table 3.2. SBERT training hyper-parameters for the NLI task

Thus, between all articles that reach the threshold, we used the following rules to compute the final score :

- An article that entails the input tweet is equivalent to 100 points;
- A neutral article is equivalent to 50 points;
- An article that contradicts the input tweet is equivalent to 0 point.

The score returned by DS-Fact is the average of the obtained points. The score is returned as a percentage. If none of the received articles reach the threshold, the module returns a score of 50%.

3.1.2.2. DS-Source. The following module is DS-Source. This module manages the database that contains the legitimacy score and the history of all the users whose post was given to the system as input. In section 3.1.2, we saw how this score is computed with the call of the update() function. DS-Source is called when the system needs to update or receive the score of a user/source.

Fig. 3.5. DS-Source Module

If the request concerns a news user, the module returns a score of 50%. Figure 3.5 presents an illustration of this module.

3.1.2.3. DS-Entity. The third module is DS-Entity. The functionality of this module is quite similar to the DS-Fact module. The only difference is the provenance of the content from which the data stream is received. Here, the sources are the Twitter accounts of the named entities mentioned in the input post. Thus, instead of getting news articles from a fact-checker website, for this module, the retrieved data are tweets.

Figure 3.6 presents the architecture of DS-Entity. The first step, once an input text is received, is to detect all the named entities in it. The upcoming steps are: get the closest tweets related to the input text that has a text similarity score equal to or above the threshold of 0.8; determine the inference of these closest tweets; compute and return the percentage by following the same rules as presented in section 3.1.2.1. The implementation of those last steps follows the same idea as for the DS-Fact module. So, we are only going to present the first step which is the named entity recognition task.

Nowadays, there are well-known tools which have demonstrated their effectiveness on the named entity recognition task on diverse applications such as Spacy, Bluemix (an IBM Watson's Cloud Platform), NLTK, StanfordNLP, Gate and OpenNLP [**72**]. Based on a study made by Dawar *et al.* [**19**], we chose to use the SpaCy tool for our system. SpaCy is

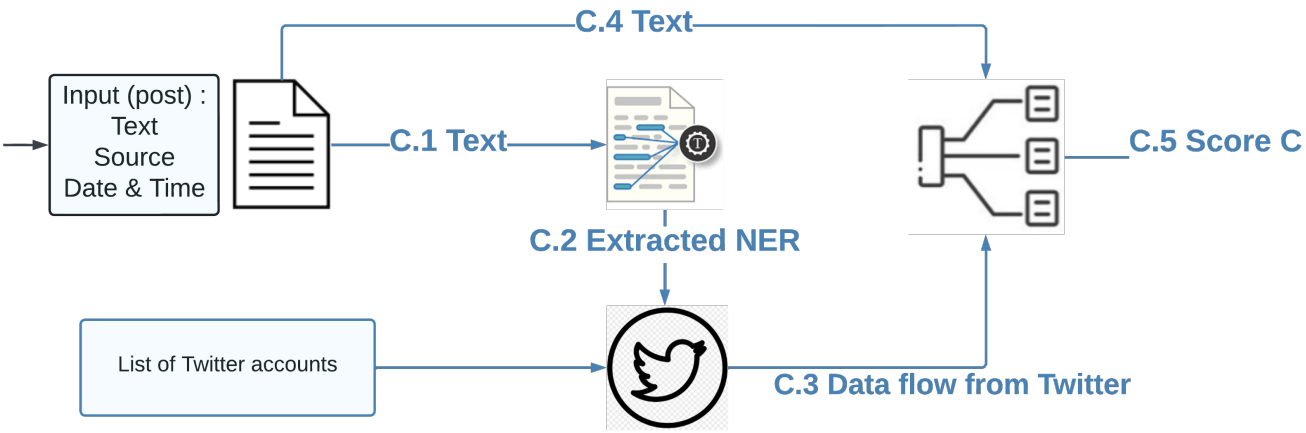


Fig. 3.6. DS-Entity Module

a free, open-source library provided by Explosion AI and designed to perform a variety of Natural Language Processing tasks.

The exact architecture of the SpaCy named entity recognition tools is not publicly available. However, we know that it employs a hybrid of Hidden Markov Models (HMMs), Maximum Entropy Models (MEMMs), and Decision Tree Analysis. All of these models are covered by a convolutional neural network to manage large and diverse datasets. HMMs are generative models for assigning joint probabilities to paired label sequences and observations. The parameters are then trained to the training sets' joint likelihood. MEMMs are conditional probabilistic sequence models that can describe several aspects of a word and can handle long-term dependency [**76, 19**].

Spacy extracts different named entities and regroup similar types of entities into labels such as GPE (for countries, cities, states, etc.), PERSON (for named person or family), ORG (for organizations, companies, agencies, institutions, etc.), DATE (for dates), LOC (for locations) and MONEY (for Monetary values, including unit), etc. Our system only keeps three labels: ORG, PER, and GPE.

Thus, DS-Entity extracts all the named entities from the input text using Spacy. Subsequently, this module starts receiving data streams from the Twitter accounts of the extracted named entities. Therefore, DS-Entity needs a database with named entity Twitter accounts. In order to create the database, upstream, we manually searched and added official and certified Twitter accounts of the top 100 named entities who have been the most mentioned in our dataset. Table 3.3 shows the 10 first elements of the dataset.

#	Named Entity	Twitter username
1	Donald Trump	@realDonaldTrump
2	Hillary Clinton	@HillaryClinton
3	Barack Obama	@BarackObama
4	Bernie Sanders	@BernieSanders
5	The white house	@WhiteHouse
6	Ted Cruz	@tedcruz
7	FBI	@FBI
8	Mike Pence	@Mike_Pence
9	Marco Rubio	@marcorubio
10	Bill Clinton	@BillClinton

Table 3.3. List of the 10 first most mentioned named entity

Our fined-tuned SBERT model for the text similarly task, as presented before is also used here to find the right named entity that was mentioned. Once more, a threshold of 0.8 is used. In some cases, we can have a match of more than one named entity. For example, if a post mentions the name "Clinton", the system will start receiving tweets from both Bill Clinton and Hilary Clinton.

We receive data streams using the Twitter API, except for Donald Trump, whose official account was suspended. Therefore, we used a dataset provided by Harvard Dataverse [**5**] that contains tweets from his suspended account. If the input tweet does not contain any named entity or we couldn't retrieve any relevant that reaches the threshold, the module returns a score of 50%.

3.1.2.4. DS-Explain. The last module that we are going to present is DS-Explain. This module is in charge of two tasks. The first is to compute the final percentage after receiving the score from the three previous modules. The second task is to generate the explanation that will be returned. Figure 3.7 shows an illustration of this module.

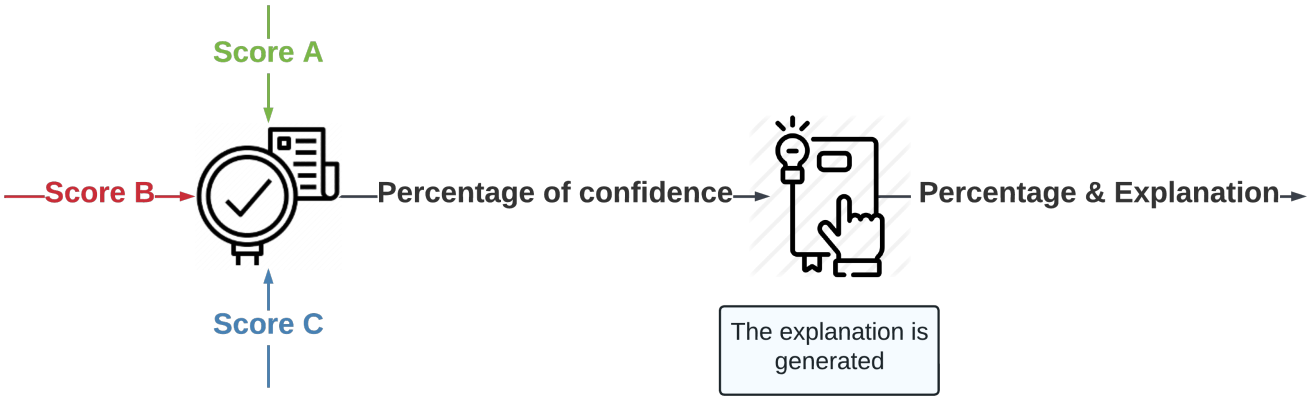


Fig. 3.7. DS-Explain Module

DS-Explain computes the percentage of confidence with a neural network model. Figure 3.8 shows the architecture of the used neural network. The score from the modules DS-Fact (score A), DS-Source (score B) and DS-Entity (score C) are the inputs X_i. The inputs are multiplied by the neuron weights W_i. After that, a bias b is added to those products. The sum Z of the result of the calculation of each input is then passed to an activation function. The activation function $f(z)$ used here is a sigmoid function. The mathematical representation of a sigmoid is as follows:

$$f(z) = \frac{1}{1 + e^{-x}}$$

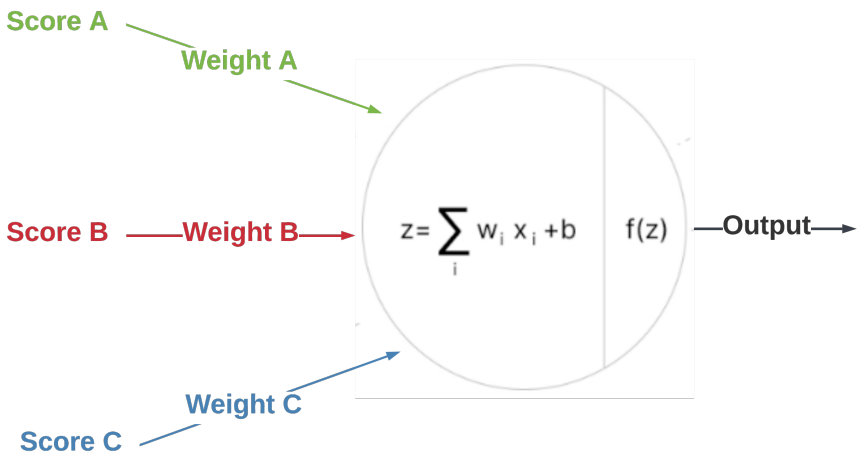


Fig. 3.8. DS-Explain Neural Networks

The result of this activation function is a value between 0 and 1. It is then transformed into a percentage, the percentage of confidence.

This module also generates an explanation that will be returned at the end of the third and last timesteps along with the final percentage of confidence. This explanation can have up to two parts. An explication can come from two of the previous modules (DS-Fact and DS-Entity). Regarding the modules DS-Fact and DS-Entity, an explanation is returned based on the last NLP task of each of those modules which is the natural language inference.

Ribeiro *et al.* [**67**] introduced an explanation method named Lime (Local Interpretable Model-agnostic Explanations). Lime as described by the authors is an algorithm that can faithfully explain the predictions of any classifier or regressor by approximating it locally using an interpretable model. As we can see in figure 3.9, an example from the original paper, Lime highlights in the text the most relevant words that contribute to predicting the obtained result. Thus, this example provides an explanation of individual predictions made by rival classifiers attempting to discern whether a document is about "Christianity" or "Atheism". The bar graph depicts the weight given to the most important words, which are also highlighted in the text. The colour of the word indicates which class it belongs to (green for "Christianity," magenta for "Atheism").

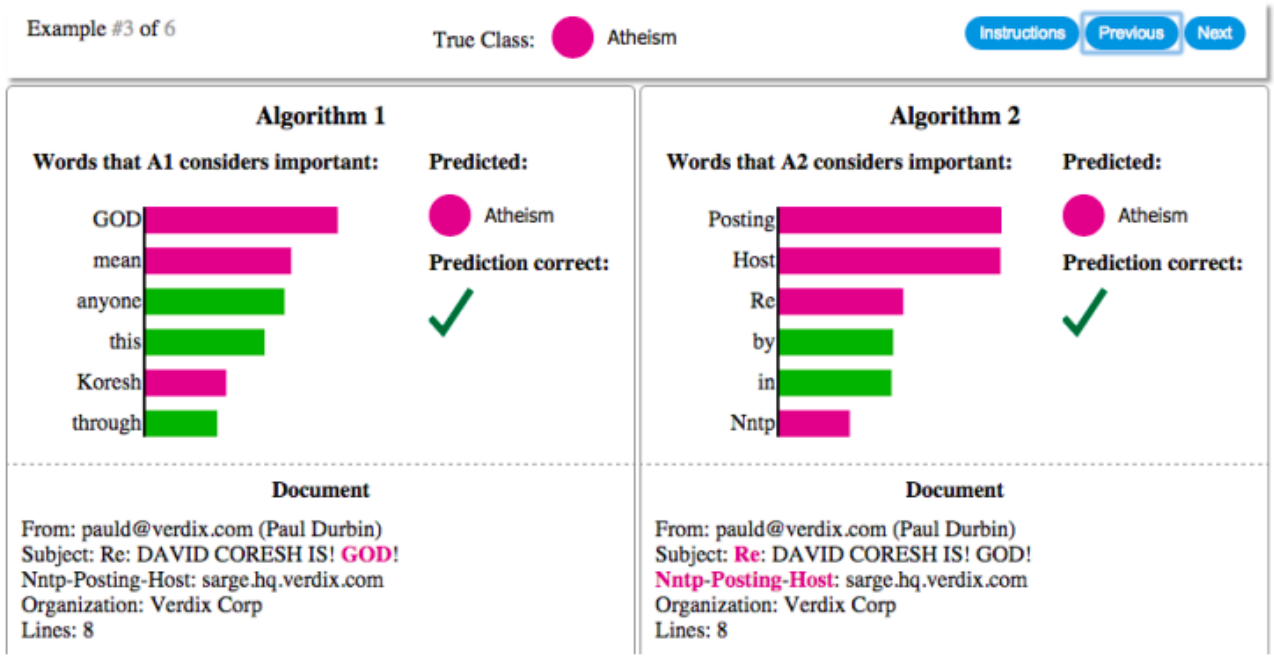


Fig. 3.9. An example of a LIME explanation

To achieve our goal of providing an explanation based on the NLI task, we followed a similar idea as Lime. Thus, the objective is to find the most important words which led to the prediction. The approach of our algorithm is as follows: We start by removing a word from the hypothesis sentence. Then we observe how the prediction score varies. The more the variation is higher the more we are certain that that word is essential. Once we have the list of important words to highlight, we present the explanation as shown in section 3.1.1.

We wanted to make that explanation clear and simple so that it can help users adopt a certain behaviour on social networks. In opposition to other explainers that only highlight words within the input text, our explication consists of returning texts from trusted articles and legitimate tweets. We highlight words within the return articles and tweets. In addition, we provide the source of the returned texts and the published date and time. Moreover, users of the DS-Fake have access to links to these articles and tweets. Thus, for example, if a user is faced with a piece of news on Twitter that reports a speech from Justin Trudeau, before sharing the information, the user will have the reflex to consult Justin Trudeau's official Twitter account. This module can return up to two articles and two tweets (the closest ones).

3.2. Conclusion

This chapter addressed the DS-Fake architecture, the first system based on data stream mining for the fake news detection problem. It also presented how particular our proposed explainer is compared to other works that provide an explanation. The last chapter showed that most works consist of finding and highlighting words that impacted the most on the obtained result. With our explainer, we wanted to help users to adopt a certain behaviour. Thus, the module that generates the explanation returns articles from fact-checkers and tweets from named entities mentioned in the input. This module also finds and highlights words but does it on the retrieved articles and tweets. On top of that, information about the source is returned as well as the published date and time and the hyperlink to the returned articles and tweets that impacted the most the obtained result (the hyperlink allows the user of the system to inspect them by himself). The idea is to show how easy it can be to verify certain information just by going to the Twitter account of the mentioned person within the post or by checking a trusted website.

Furthermore, this chapter also presented how the legitimacy score is calculated. While most work use features such as the number of people the user is following or his writing style to assign a credibility score to a source, the DS-Fake system use the post history. Only a few papers use the post history, and when it comes to computing a source's credibility score,

only previous posts from that source are considered. Our legitimacy score includes previous posts from the source and previous posts from all sources using the Bayesian average. This average, in proportion to the number of posts in the history, lessens the impact of the results from earlier posts.

This chapter also presented for the first time an approach that encapsulates the text similarity task and the natural language inference task for this problem. The next chapter aborts our experimentation and the obtained results.

Chapter 4

Experimentation

The present chapter mainly focuses on the experimentation part of our work on the DS-Fake system. The first part presents the different metrics used to evaluate our work. The second part will address the experiments that have been made and the obtained results. To highlight the performance of our data stream mining approach, we used the metrics presented in the paper that introduced the FakeNewsNet dataset and the state-of-the-art baselines on this dataset and the same subset. Additionally, this chapter presents the performance of the modules of DS-Fake, which include the module that manages the legitimacy score based on the Bayesian average and the module that combines the text similarly and the natural language inference tasks.

4.1. Evaluation metrics

This section is on the evaluation metrics. We will present four well-known metrics that we used to evaluate our work. However, DS-Fake returns a percentage that we named the percentage of confidence which tells how likely we can trust the content of the input tweet. The dataset that we are using, as presented in section 2.2.1, gives one of the following labels to reviewed tweets: pants on fire, false, barely true, half true, mostly true, and true. Thus, we had to find a way to map the percentage of confidence to the labelling of the dataset. Table 4.1, shows how we did the mapping. Here, we are in a multi-class classification and if, for example, an input post receives a percentage of 75%, we can conclude that the post was attributed to the class mostly true.

The four metrics that we are going to use are the macro-accuracy, the macro-recall, the macro-precision and the macro-F1. However, before presenting those metrics for our multi-class classification, we have to introduce the following terms in a binary classification task: accuracy, precision, recall and precision.

Dataset labels	DS-Fake percentage of confidence
True	[87% - 100%]
Mostly true	[70% - 86%]
Half true	[53% 69%]
Barely true	[37% - 52%]
False	[21% - 36%]
Pants on fire	[0% - 20%]

Table 4.1. Mapping from the dataset labelling to the percentage of confidence

In binary classification, the class of an input can be either 0 (Negative) or 1 (Positive). To visualize the performance of a classifier, the best way is to use a confusion matrix. A confusion matrix, as we can see in Table 4.2, is a cross table that keeps track of the number of occurrences between two raters, as well as the true/actual classification and the predicted classification. [**29**]

Confusion Matrix		Predicted	
		0	1
Actual	0	True Negative (TN)	False Positive (FP)
	1	False Negative (FN)	True Positive (TP)

Table 4.2. Confusion Matrix for Binary Classification

The confusion matrix of a binary classification has two rows and two columns. The first row shows how many negative samples were predicted as positive or negative and the second row shows how many positive samples were predicted as positive or negative. Therefore, we can introduce the following terms:

- True Negative (TN): It refers to the number of negative samples that have been correctly labelled as negative.

- True Positive (TP): It refers to the number of positive samples that have been correctly labelled as positive.

- False Negative (FN): It refers to the number of positive samples that have been incorrectly labelled as negative.

- False Positive (FP): It refers to the number of negative samples that have been incorrectly labelled as positive.

We can therefore present the metrics using the previous terms. Accuracy is the ratio of samples correctly predicted among the total number of samples. It expresses the number of both positive and negative samples correctly classified. The following formula is used to compute the accuracy:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

Precision is the ratio of positive samples correctly predicted among the total number of samples predicted as positive. It expresses the number of correct positive predictions that were made. The following formula is used to compute the precision:

$$Precision = \frac{TP}{TP + FP}$$

Recall is the ratio of positive samples correctly predicted among the total number of positive samples. It expresses the number of actual positive samples correctly classified. The following formula is used to compute the recall:

$$Recall = \frac{TP}{TP + FN}$$

F1-score aggregates the precision and the recall into a single measure using the harmonic mean. It's the weighted average between precision and recall. The F1-score reaches its highest score at 1 and its lowest at 0. The following formula is used to compute the F1-score:

$$F1 - score = 2 \times \frac{Precision \times Recall}{Precision + Recall} = \frac{2TP}{2TP + FP + FN}$$

The previous formulas can also be used for a multi-class classification problem. Thus, for a multi-class, we have to compute the accuracy, the precision, the recall and the F1-score

separately for each of our classes. This comes down to finding TP, TN, FP and FN for each class individually.

Suppose we want to compute the accuracy, the precision, the recall and the F1-score for the class "Half true". Table 4.3 shows a confusion matrix for this particular class. We can use the formulas presented above. Nevertheless, the previous terms are defined as follows:

- True Negative (TN): It refers to the number of samples that don't have the label "Half true" and have been correctly labelled as not "Half true".
- True Positive (TP): It refers to the number of samples that have the label "Half true" and have been correctly labelled as "Half true".
- False Negative (FN): It refers to the number of samples that have the label "Half true" and that have been incorrectly labelled as not "Half true".
- False Positive (FP): It refers to the number of samples that don't have the label "Half true" and that have been incorrectly labelled as "Half true".

		Predicted					
	Classes	True	Mostly true	Half true	Barely true	False	Pants on fire
Actual	True	TN	TN	FP	TN	TN	TN
	Mostly true	TN	TN	FP	TN	TN	TN
	Half true	FN	FN	TP	FN	FN	FN
	Barely true	TN	TN	FP	TN	TN	TN
	False	TN	TN	FP	TN	TN	TN
	Pants on fire	TN	TN	FP	TN	TN	TN

Table 4.3. Confusion Matrix of a multi-class classification (case of the class Half true)

The same logic can be used for all classes. Once we have the accuracy, the precision, the recall and the F1-score for each class, we can merge them to compute the macro-accuracy, the macro-precision, the macro-recall and the macro F1-score.

The macro-accuracy can be computed with the following formula, where K is the number of classes and $Accuracy_k$ is the accuracy of class k:

$$Macro - accuracy = \frac{\sum_{k=1}^{K} Accuracy_k}{K}$$

The macro-precision can be computed with the following formula, where K is the number of classes and $Precision_k$ is the precision of class k:

$$Macro - precision = \frac{\sum_{k=1}^{K} Precision_k}{K}$$

The macro-recall can be computed with the following formula, where K is the number of classes and $Recall_k$ is the recall of class k:

$$Macro - rappel = \frac{\sum_{k=1}^{K} Recall_k}{K}$$

The macro F1-score can be computed with the following formula, where K is the number of classes and $F1 - score_k$ is the F1-score of class k:

$$Macro\ F1 - score = \frac{\sum_{k=1}^{K} F1 - score_k}{K}$$

4.2. Experiments and Results

This section addresses the experiments carried out as part of this work. The first type of experimentation focuses on the importance of each module of DS-Fake. The second type touches on how DS-Fake performs over the timesteps. For these two first experiments, we used the sub-dataset as presented in section 2.2.1. We also compare the result of DS-Fake with other approaches that used the same dataset. The entire dataset was used for this comparison. This section is organized as follows. We start by presenting the experimental settings. The second part of this section presents the experimentation on the timesteps. The next part addresses experimentations on different configurations of DS-Fake's modules. Finally, we show the result of our system against other works on the same dataset.

4.2.1. Experimental settings

Before addressing the different experiments, we have to present the settings used for those experiments. The following configurations are the same for the experimentations that will be presented in subsections 4.2.2 and 4.2.3.

First, let's present the dataset. We choose to use a subpart of the data due to the number of experiments to launch and the execution time of each experiment. Table 4.4 shows the repartition of the data for our experimentation.

Thus, we randomly chose a subset of the dataset that contains 15000 tweets. We partitioned this subset into three parts: the training subset, the validation subset, and the test subset. The training set is composed of 80% of the subset dataset, the validation set has 10% and the test set has the remaining 10%. Once more, we chose 150 articles that are on the same topics as the chosen subset of tweets. We partitioned them in the same proportion as tweets.

	Training subset	Validation subset	Test subset
Content of articles	120	15	15
Articles author's name	112	13	14
Article published date and time	120	15	15
Content of tweet	12000	1500	1500
Tweet author's name (username)	9812	878	941
Tweet published date and time	11868	1282	1330

Table 4.4. Experiment dataset statistics

As we can see in table 4.4, there is some missing information mostly due to the limitation of the Twitter API. There is a limited amount of information that we can retrieve daily, which significantly increased the time to collect the data. Some users have put some security restrictions on their Twitter accounts that can not allow us to access to all information about the user and his posts.

Specification	Value
CPU model	Intel (R) Xeon (R)
CPU frequency	2.30 GHz
Number of CPU cores	2
RAM (Random Access Memory)	26.30 GB
Disk space	34 GB
GPU	NVIDIA Tesla P100 - PCIE
GPU memory	16 GB

Table 4.5. Google Colab pro specifications

All experiments were done on the same computing infrastructure. We used the Google Colaboratory environment with a colab pro account. Table 4.5 describes the specifications of this environment.

4.2.2. DS-Fake timesteps

The DS-Fake system returns a percentage at three timesteps. Between these timesteps, the system receives data streams from different sources. The idea is to receive more relevant information in between the timesteps in order to return a percentage that is more trustable than the previous returned based on the latest retrieved information. Therefore, we have to find the ideal duration of each timestep that provides a good trade-off between performance and speed. The goal is to have a system that returns good results in a short among of time. Thus, in this section, we are going to test different durations.

We started our experimentation by using three different approaches. The first approach uses short-term time steps, the second uses mid-term time steps and the last approach uses long-term time steps. For each approach, we tested three different values arbitrarily chosen.

Approach	Timestep 1	Timestep 2	Timestep 3
Short-term A	0 second	60 seconds	120 seconds
Short-term B	30 seconds	90 seconds	150 seconds
Short-term C	60 seconds	120 seconds	180 seconds
Mid-term A	1 minute	5 minutes	10 minutes
Mid-term B	2 minutes	8 minutes	16 minutes
Mid-term C	5 minutes	15 minutes	25 minutes
Long-term A	25 minutes	50 minutes	75 minutes
Long-term B	30 minutes	60 minutes	90 minutes
Long-term C	50 minutes	100 minutes	150 minutes

Table 4.6. Timesteps approaches

Table 4.6 shows the different durations that we tested each approach. The time column of timestep 1 represents the waiting time between the reception of a request and the launch of the system. The time column of timesteps represent the duration of the sleep() function as presented in the pseudocode (algorithm 3.1.1) in the general architecture section (section 3.1.2).

Grandini *et al.* [**29**] presented the macro F1-score as a good indicator of how an algorithm performs on all classes. Thus, for this experience, we chose to use the macro F1-score as a metric to find the timestep configuration that helps us to find a good trade-off.

Figure 4.1 present the result of the tested short-term approaches. The time range for this type of approach is between 0 and 180 seconds after the request is received. The lowest macro F1 score is 58% and the highest score that we get is 65%. It can be observed that trying to return a percentage shortly after a request does not yield an acceptable result. Furthermore, the score does not increase significantly.

The result of the experiment on the mid-term approaches is presented in figure 4.2. Here, the time range is between 1 minute and 25 minutes. The lowest macro F1 score is 58% and the highest is 85%. Even though we have to wait until the third time step

Fig. 4.1. Macro F1-score of the short-term approaches

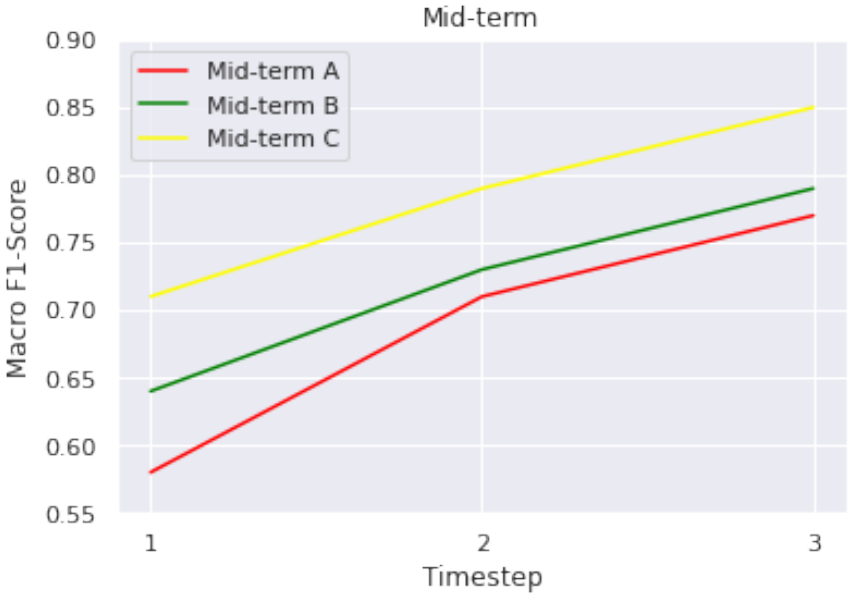


Fig. 4.2. Macro F1-score of the mid-term approaches

of the mid-term C, with this family of approach, we can observe that the score increases significantly with the timesteps. 85% is the highest score we get with DS-Fake in all the experimentations and tests we did.

Figure 4.3 shows the outcome of the experiment on the long-term approaches. The time range is between 25 minutes and 75 minutes. As we can see, the three lines are

confused. They overlap each other. Thus, the score does not change between the timesteps of all tested values of this family. The score starts and remains at 85%. This phenomenon occurs because between 25 and 75 minutes after the beginning of the process, DS-Fake didn't find relevant posts or articles that reached the required threshold to impact the returned input' percentage of confidence.

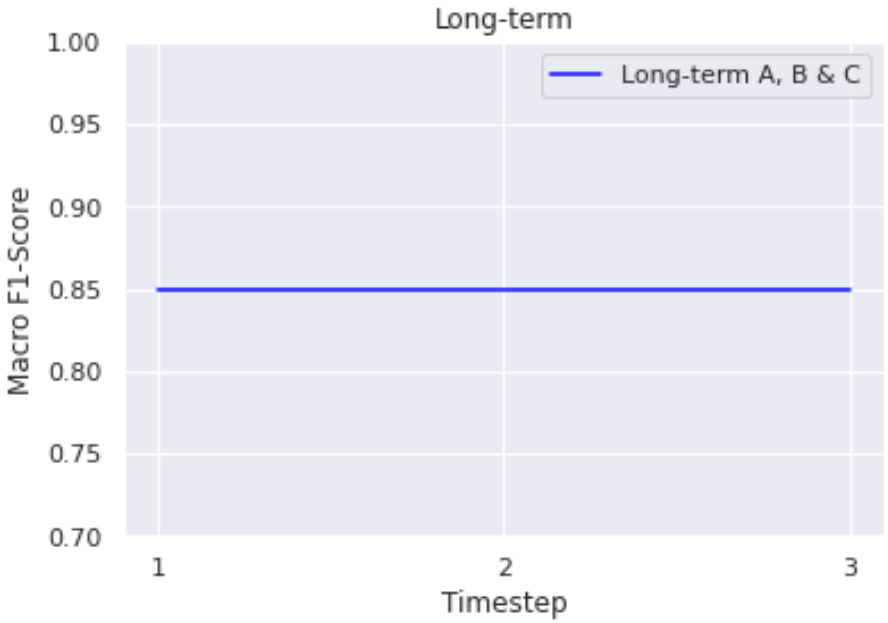


Fig. 4.3. Macro F1-score of the long-term approaches

The above observation leads us to test a new family of approaches combining the short-term and the mid-term approaches. We did not include the long-term approach because the result of this approach demonstrated that beyond 25 minutes, the result does not increase. This can be justified by the way our dataset was created. The dataset is composed of tweets related to published news articles. These articles are verified by the fact-checkers website shortly before or after the tweet is posted. Thus, further investigation of our system will not lead to better results.

Regarding the short-term approach, we noticed that returning the percentages over a short period works but does not provide high performance. Our hypothesis to explain this behaviour is that DS-Fake receives a significant amount of data streams. The system has to check the similarity and inference of each of the retrieved articles and posts, which can be time-consuming. The short-term approach does not give enough time to our system to do all the computation and return better results.

The mid-term approach, as we observed, provides enough time for our system to do the calculation improves the score with the timesteps. Nevertheless, our goal is to have a good trade-off between performance and speed. Consequently, we also tried different values of the mix of the mid-term and short-term approaches. Table 4.7 shows the chosen timesteps for our system after testing different values of this fourth family of approaches.

Approch	Timestep	Value
Mix-term	Timestep 1	2 minutes
	Timestep 2	10 minutes
	Timestep 3	20 minutes

Table 4.7. DS-Fake timesteps

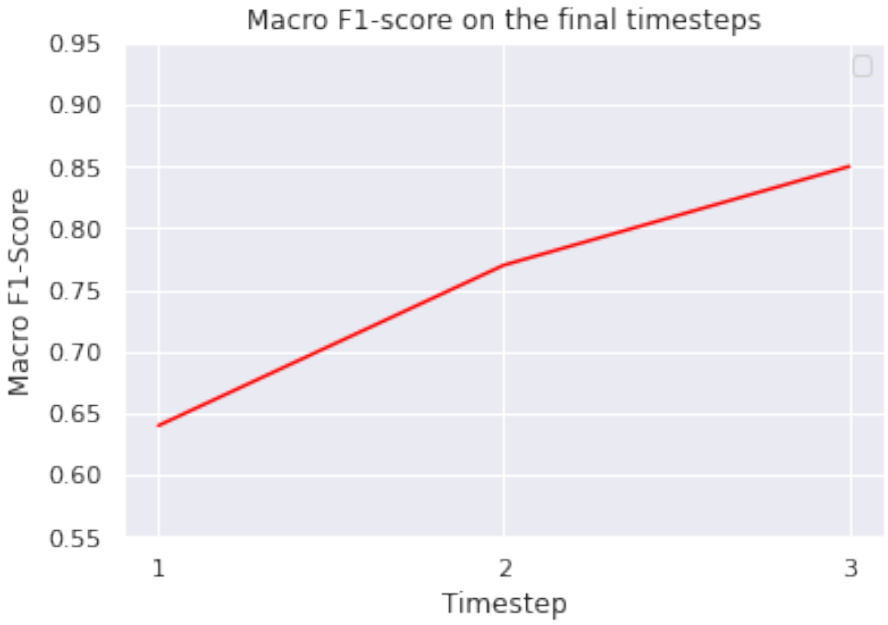


Fig. 4.4. Macro F1-score on the final timesteps

Figure 4.4 presents the macro F1-score of the chosen timesteps. We can observe a line that is almost straight that starts with a score of 64% and end with a score of 85%.

4.2.3. DS-Fake modules

The second experiment we carried out was to determine the impact of each of the DS-Fake modules. We wanted to have a better understanding of each module of our system. In order to reach that goal, we tested different combinations of modules.
All our experiments were done on the data presented in section 4.2.1 and with the time steps discussed in section 4.2.2. The different combinations are summarized in Table 4.8.

These combinations do not include the part of the DS-Explain module that is in charge of generating the explanation. When DS-Explain is used in the following combination, we only consider the neural network that computes the percentage returned by DS-Fake.

DS-Source is based on the history of each user. The legitimacy score returned by this module does not depend on any computation. So, for experimentations that only include DS-Source or DS-Source with DS-Explain, we assumed that whenever a request posted by a certain user/source is received, DS-Source returned the legitimacy score as usual. However, when it's time to update the database, we consider that DS-Fake has found the correct class and the user's history has been updated accordingly.

Combination	Module			
	DS-Fact	DS-Source	DS-Entity	DS-Explain
Combination 1	X			
Combination 2		X		
Combination 3			X	
Combination 4	X			X
Combination 5		X		X
Combination 6			X	X
Combination 7	X	X		X
Combination 8		X	X	X
Combination 9	X		X	X

Table 4.8. DS-Fake modules combinations

The first combinations consist in simply using one of the three modules which are supposed to run simultaneously (DS-Fact, DS-Source and DS-Entity). We didn't consider the module DS-Explain alone because this module can not work without at least one of the previously listed modules. Figure 4.5 shows the macro F1-score of the combinations 1,2 and 3.

Combination 1 with only the DS-Fact modules outperforms the first three combinations with a final score at timesteps 3 of 72%. We also tested different timesteps with only DS-Fake. We noticed that the result of this module is directly proportional to the duration of the timesteps. The score increases when the duration of each time step increase and vice versa.

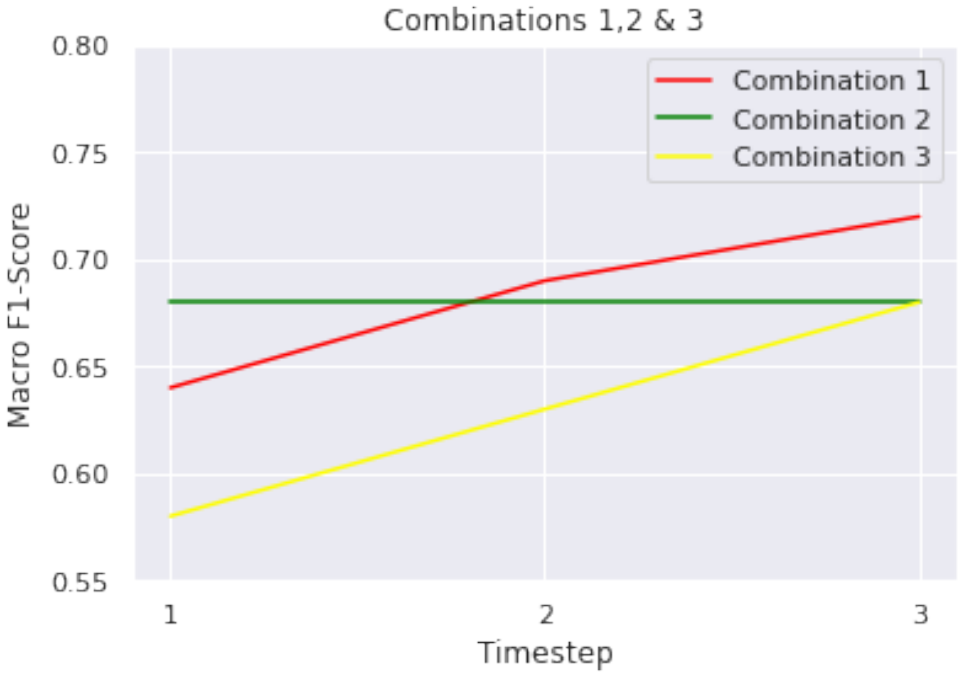


Fig. 4.5. The Macro F1-score of combinations 1, 2 and 3

Secondly, as expected, the score of the combination 2 with DS-Source only does not change through the time steps. The legitimacy score of a user/source only changes after the final result has been computed. When the system is executing a query, the legitimacy score remains the same.

On the other hand, we were expecting a higher score than the 68% obtained when DS-Source is executed alone. Our hypothesis to explain this result was on the number of times that the system received posts from a particular user/source because DS-Source works on the history of each user/source. Thus, we explored our dataset to find out the

occurrence of posts received for each user/source.

Figure 4.6 shows the percentage of users/sources with a certain number of posts. We can see, for example, that only 1.3% of users/sources have more than 50 posts and 21,3% of users/sources have between 11 and 50 posts. So, the obtained result can be explained by the fact that 35,3% of users/sources can not have a history since they only have one post.

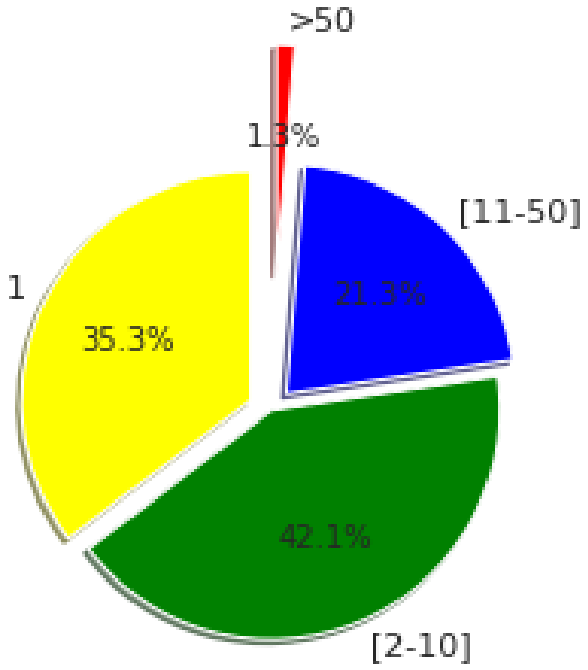


Fig. 4.6. The percentage of users/sources with a certain number of posts

The third combination with only DS-Entity had the worst score at timestep 1 and finished at timestep 3 with a score of 68%, tied with combination 2. Once again, we explored our dataset to understand the result.

Our assumption was based on the presence of named entities on our dataset because DS-Entity operates on data streams received from named entities on the input post. We discovered that 85.7% of tweets contain at least one named entity. Even though the percentage is relatively high, we did not get a score slightly close to the combination 1 that also retrieves streams data. Additional investigations were conducted to understand this result. Thus, we can conclude that the result is lower than expected because we found out that our database with official and certified named entity Twitter accounts is not inclusive. There are named entities that are not listed there. In addition, between those present in the database, there are a certain number of named entities that rarely tweet.

Combinations 4, 5 and 6 are slightly the same as the first three combinations, except that we've included the DS-Explain module at the end of each previous module. Figure 4.7 presents the macro F1-score of these combinations.

We can observe that the DS-Explain module improved the result of all the modules (DS-Fact, DS-Source and DS-Entity). Even if the increase is not huge, adding a neural network helps to have better results.

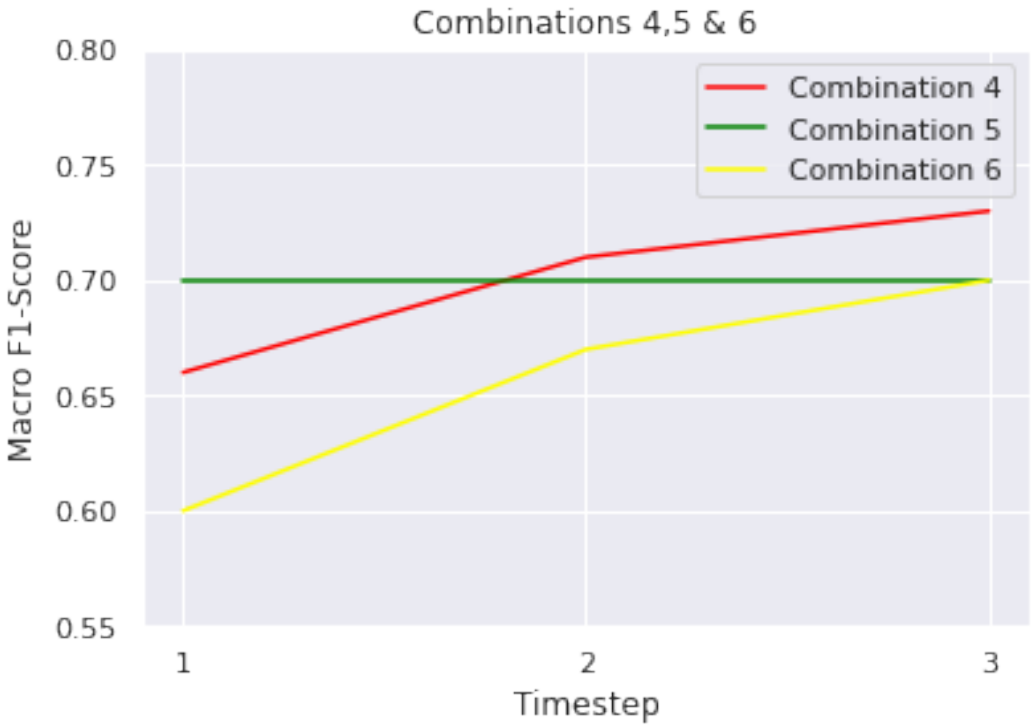


Fig. 4.7. The Macro F1-score of combinations 4, 5 and 6

The last three combinations are composed of a mixture between two of the modules that run in parallel and the DS-Explain module. We could not test those mixtures without the DS-Explain module because we needed it to compute the percentage returned. Each module, whether DS-Fact, DS-Source or DS-Entity, returns a score. So, we need the neural network to combine these results to return one percentage. Figure 4.8 presents the macro F1-score of these combinations.

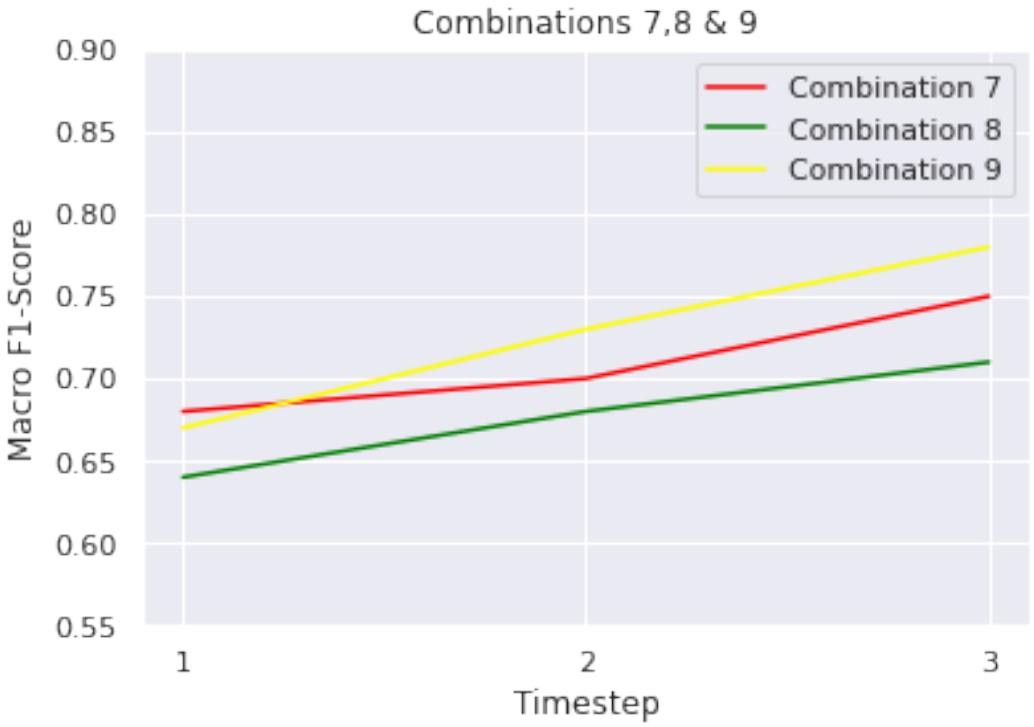


Fig. 4.8. The Macro F1-score of combinations 7, 8 and 9

Combination 7, which mixes the DS-Fact and the DS-Source modules, had the highest score at timestep 1 but finished in the second place at timestep 3 with a score of 75%. Combination 8, which mix the DS-Source and the DS-Entity modules, had the worst score at all timesteps. This behaviour was expected since this combination merges the modules that performed the worst when used alone or with DS-Explain. Combination 9 outperforms all the presented combinations with a score of 78% at timestep 3. Once more, we expected this result since it regroups the DS-Fact and the DS-Entity modules. These modules returned the best results when used alone or with DS-Explain.

This experimentation allowed us to understand the impact of each module. We clearly saw that to have the best performance, we need to include all the four modules of DS-Fake (DS-Fact, DS-Source, DS-Entity and DS-Explain). As seen before, since we are mixing modules, DS-Explain is important because it help us to merge different scores into a single percentage. Regarding the modules that execute in parallel, even though all of them are important, with our dataset, we observed that the DS-Fact module has the greatest impact followed by the DS-Entity module. However, this order will probably change depending on the used dataset, but our system will still work well because each module completes the lack of others. If, for example, we have a dataset where there are not enough named entities that can be extracted, but we have access to a complete history of a significant number of

users/sources, then DS-Source will have better importance than DS-Entity.

4.2.4. DS-Fake vs benchmark

In this subsection, we are going to present the performance of our DS-Fake system compared to the state of the art as introduced by the author of the FakeNewsNet dataset.

Shu *et al.* [**79**] used three categories of approaches as the state-of-the-art baseline for this particular benchmark because they wanted to cover all types of features in the FakeNewsNet dataset. As presented in table 2.5, this dataset has news content features, social content features and spatiotemporal features.

For the news content features, authors represented the text content as a one-hot encoded vector, then conventional machine learning models such as support vector machines (SVM), logistic regression (LR), Naive Bayes (NB), and Convolutional Neural Network (CNN) were used for the evaluation.

In addition, the authors also used a model that was introduced by Shu *et al.* [**78**]. This model named Social Article Fusion (SAF), employs an autoencoder to learn features from news articles in order to classify news articles as fake or real. This model has three versions, detailed as follows:

- SAF/S which only uses the news article content. It detects fake news based on the input text.

- SAF/A which only uses the news article context. It detects fake news based on the temporal pattern of user engagements.

- SAF which uses both news article content and context. It's a combination of SAF/S and SAF/A.

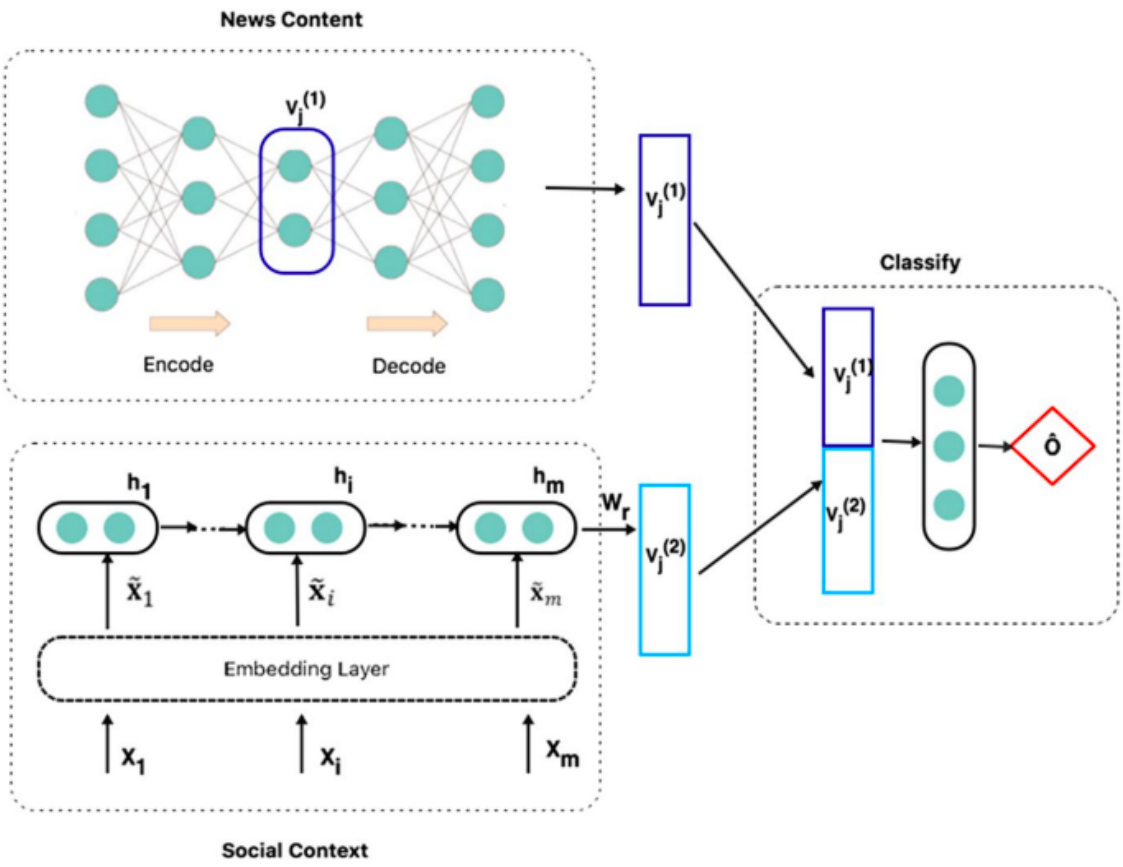


Fig. 4.9. The general architecture of the Social Article Fusion model

All three versions of the SAF model were used for the evaluation. This model employs an autoencoder with two-layer LSTM cells for encoder and decoder, as well as a network of two-layer LSTM cells to record the temporal pattern of user engagements. Figure 4.9 shows the general architecture of the Social Article Fusion model (SAF).

Table 4.9 presents the result of the state-of-the-art baselines (as described at the beginning of this subsection) of the FakeNewsNet dataset and the DS-Fake system on this same dataset and the same subset. The same metrics introduced by the authors of this dataset were used here. These metrics, which are identical for all experiments, are detailed in section 4.1 (Macro-accuracy, macro precision, macro recall and macro F1-score).

The macro-accuracy of the baselines remains more or less close to 0.65 while DS-Fake could achieve 0.808. The macro-precision of the baselines ranges from 0.600 for the SAF /S to 0.807 for the CNN method. Our system slightly exceeded this score with a macro-precision of 0.841. Among the baseline approaches, the SAF got 0.789 which is the best macro-recall on par with the SAF /S. The CNN approach with a score of 0.456 has the worst macro-recall score. The proposed system reaches a score of 0.871. The highest

Model	Metric			
	Macro-accuracy	Macro-precision	Macro-recall	Macro F1-score
SVM	0.580	0.611	0.717	0.659
Logistic regression	0.642	0.757	0.543	0.633
Naive Bayes	0.617	0.674	0.630	0.651
CNN	0.629	0.807	0.456	0.583
Social Article Fusion /S	0.654	0.600	0.789	0.681
Social Article Fusion /A	0.667	0.667	0.579	0.619
Social Article Fusion	0.691	0.638	0.789	0.706
DS-Fake	**0.808**	**0.841**	**0.871**	**0.855**

Table 4.9. Fake news detection performance on FakeNewsNet

performance increase is on the macro F1-score. DS-Fake got a score of 0.855. The method that got the worst macro F1-score between the baselines is the CNN method with a score of 0.583. The best method between the baselines is the SAF, which got 0.706.

DS-Fake outperforms all the state-of-the-art baselines of this FakeNewsNet benchmark on all the metrics.

4.3. Conclusion

This chapter was about the experiments that have been made on the proposed system. The first part introduced the metrics used for the evaluation. The second part addressed the experimentations and the obtained results. The same metrics used by the FakeNewsNet dataset authors' to present the results of the state-of-the-art baselines were used in this chapter. All experiments were done on this dataset, and DS-Fake outperforms all state-of-the-art baselines on all parameters. Thus, this chapter showed us that the data stream mining approach achieves good results.

Furthermore, the experimentation made to find out the impact of each module helped us to highlight the performance of two other approaches that we proposed for the fake news detection problem, which are the legitimacy score and the combination of the two NLP tasks. When used alone, the module that return the legitimacy score based on the Bayesian average had an F1 score of 68%. This score is better than five of the seven state-of-the-art baseline approaches. This behaviour is the same for the F1-score at the third timesteps for the modules that encapsulate the text similarity task and the NLI task.

Chapter 5

Conclusion and future work

This thesis presented the DS-Fake system. A near real-time data stream mining based approach against fake news. This system is composed of four modules that work together to return the percentage of confidence of an input tweet at three time steps. The higher the percentage, the more the tweet can be trusted. In this work, the execution of this system and each module is simulated as if it was a real-life situation.

The first module, named DS-Fact, is in charge of computing a score based on the received data flow from trusted sources such as fact-checker websites. This module retrieves the articles closest to the input which are published or verified by fact-checkers. Then a score is computed depending on the number of articles that infers the input post over the time among the closest retrieved articles. An SBERT pre-trained model was fine-tuned for the text similarity and the NLI task.

The second module was named DS-Source. This module manages the legitimacy score of Twitter users whose tweets have been given as input to the system. We chose to use the post history instead of assigning a score to a user based on features like his localization, job title, number of followers, etc. In recent literature, rare are the papers that used the post history with success. And when history is used, a score is given to a user based on the results average of his previous posts. Our credibility score named the legitimacy score assigns a score to a user based on his previous posts and the previous posts of all the users using the Bayesian average. Depending on how many posts have been made in the past, the Bayesian average reduces the impact of the results of previous posts. A user who has over one hundred posts in the history that are all true is more credible than a second user with two posts that are all true as well. His legitimacy score will therefore be higher than that of the second user. Thus, once the final percentage of an input post is calculated, the score of all the users is updated. Experimentation on this module has shown that the legitimacy

score, when used alone, performs better than most state-of-the-art baselines.

The third module is DS-Entity. This module is built in the same way as DS-Fact. However, the data used here comes from a data stream from the official Twitter accounts of the named entities present in the input tweet extracted using the Twitter API. This system also has a manually created list of official Twitter accounts of the most mentioned named entities of the used dataset. We used the spacy tool to extract the named entities.

The first task of the last module, named DS-Explain, is to merge the scores of the three previous modules and compute the returned percentage with the help of a neural network model. The second task of this module is to generate an explanation. Unlike most systems that include an explainer that finds and highlights important words within the input text, we wanted to have a better explainer that can help users of our system to develop a particular behaviour when faced with information on social networks. Thus, DS-Explain returns the text from the trusted and legitimate sources and highlights words that influenced the most the obtained result within these trusted and legitimate texts. A trusted source is an article from a fact-checkers website, and a legitimate source is a post from a named entity mentioned in the input text. This module also returns the information about the sources, the published dates and times and links to the articles or the tweets. The purpose is to demonstrate how simple it is to confirm certain information by visiting the mentioned named entity's Twitter account or a reliable website.

To the best of our knowledge, the DS-Fake system is the first that based its approach on data stream mining. It outperforms all the state-of-the-art baselines introduced by the authors of the FakeNewsNet dataset. The experiments were done using the same dataset and the same metrics. The proposed system showed a significant performance increase on all metrics. We also conducted experiments that helped us understand the impact of each module and determine the best timing for the time steps. This thesis proves that receiving a flow of data through time instead of returning a result based on the available data when a request is received can also be a good approach to solving the fake news detection problem in a real-life situation.

In addition to the explanation provided by the system and the improvement of the state-of-the-art model results, this work also has other contributions that we want to highlight. This work is the first to propose a model that encapsulates the result of the text similarity and the natural language inference (NLI) task for the fake news detection task. Our experiments also showed that this encapsulation performs better than most state-of-the-art baselines. Furthermore, our work contributes to enriching the diversity of

the FakeNewsNet dataset. The legitimacy score and the receive data stream from identified named entities within the input enhance this dataset with new social context elements.

5.1. Future work

The DS-Fake system showed significant results in terms of performance, findings and contributions for the fake news detection problem. Despite this, the fight against fake news requires ongoing research mainly due to the growing scale of this phenomenon. Thus, this system can always be improved, and more experiments can be made. The first stage of our future work is presented in the four following points.

The first point regards the architecture of DS-Fake. The current version of this system only includes textual input. The idea is to transform it into a multimodal system that can receive as input videos and images. Thus, we will add some techniques such as OCR (Optical Character Recognition) to be able to extract the contents of an image.
Furthermore, we will try other existing approaches for text similarity, named entity recognition (NER) and natural language inference (NLI). The goal is to conduct experimentation with different approaches on different datasets to find out which one performs the best on a large variety of datasets. The DS-Fake system will therefore be more inclusive.

The second point touches on the dataset. As we mentioned in the previous point, we have to test our system on different datasets. We are working on finding data about politics but also on different topics. In addition, we want to include more fact-checkers websites, such as FactCheck.org or Snopes.com. On top of fact-checkers, we want to add other types of trusted sources like websites of well-known media such as CNN or CBC.

The next point focuses on the DS-Source module. First, we want to push our experimentation further in order to find out the minimum number of posts from a user/source necessary to have a legitimacy score that best represents the "fake news history" of a user/source. Secondly, we want to relativize this score according to the topics. For example, a user may be an expert on politics. His tweets about politics have a percentage of 100%. However, when it comes to other topics like physics, the average percentage of the user's tweets is around 30%. Thus, in other words, we want to have an overall legitimacy score and also sub-legitimacy scores which will be based on particular topics.

The last point is on the DS-Explain module, or more particularly the part of this module that generates the explanation. The idea behind was to help users to realize and to increase their ability to spot fake news. We want users to adopt a certain behaviour on online social media. Thus, we are going to do a survey of users of the DS-Fake system. This survey will allow us to see if our module achieves its objective and to improve it, if necessary, with the help of user feedback.

References

[1] Algolia : Using the bayesian average in ranking.

[2] N. Ashraf, S. Butt, G. Sidorov et A. Gelbukh : Cic at checkthat! 2021: Fake news detection using machine learning and data augmentation. *CEUR Workshop Proceedings*, 2936:446–454, 2021.

[3] S. Auer, C. Bizer, G. Kobilarov, J. Lehmann, R. Cyganiak et Z. Ives : Dbpedia: A nucleus for a web of open data. *The Semantic Web*, pages 722–735, 2007.

[4] B. Bamiro et I. Assayad : Data-based automatic covid-19 rumors detection in social networks. *Proceedings of the 4th International Conference on Networking, Information Systems Security*, 2021.

[5] J. Baumgartner : Twitter Tweets for Donald J. Trump (@realdonaldtrump), 2019.

[6] G. Bekoulis, C. Papagiannopoulou et N. Deligiannis : A review on fact extraction and verification. *ACM Computing Surveys*, 55, 2021.

[7] S. R. Bowman, G. Angeli, C. Potts et C. D. Manning : A large annotated corpus for learning natural language inference. *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 632–642, 2015.

[8] J. Bromley, I. Guyon, Y. LeCun, E. Sickinger et R. Shah : Signature verification using a "siamese" time delay neural network. *Advances in Neural Information Processing Systems 6*, 1993.

[9] C. Budak : What happened? the spread of fake news publisher content during the 2016 u.s. presidential election. *The World Wide Web Conference*, 363(6425):139–150, 2019.

[10] A. Burston, J. C. Barrios, D. Gomez, I. Sturm, A. Uppal et Y. Yang : A brief history of fake news.

[11] E. Canhasi, R. Shijaku et E. Berisha : Albanian fake news detection. *ACM Trans. Asian Low-Resour. Lang. Inf. Process.*, 2022.

[12] C. Castillo, M. Mendoza et B. Poblete : Information credibility on twitter. *Proceedings of the 20th International Conference on World Wide Web*, page 675–684, 2011.

[13] D. Cer, M. Diab, E. Agirre, I. Lopez-Gazpio et L. Specia : Semeval-2017 task 1: Semantic textual similarity multilingual and crosslingual focused evaluation. *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 1–14, 2017.

[14] M. Chalkiadakis, A. Kornilakis, P. Papadopoulos, E. Markatos et N. Kourtellis : The rise and fall of fake news sites: A traffic analysis. *13th ACM Web Science Conference 2021*, page 168–177, 2021.

[15] M. Chen, X. Chu et K. P. Subbalakshmi : Mmcovar: Multimodal covid-19 vaccine focused data repository for fake news detection and a baseline architecture for classification. *Proceedings of the 2021 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, pages 31–38, 2021.

[16] C. Clifton : Data mining.

[17] N. Corbu, D. Oprea, E. Negrea-Busuioc et L. Radu : 'they can't fool me, but they can fool the others!' third person effect and fake news detection. *European Journal of Communication*, 35(2):165–180, 2020.

[18] M. Davoudi, M. R. Moosavi et M. H. Sadreddini : Dss: A hybrid deep model for fake news detection using propagation tree and stance network. *Expert Systems with Applications*, 198, 2022.

[19] K. Dawar, A. J. Samuel et R. Alvarado : Comparing topic modeling and named entity recognition techniques for the semantic indexing of a landscape architecture textbook. *2019 Systems and Information Engineering Design Symposium (SIEDS)*, pages 1–6, 2019.

[20] A. De, D. Bandyopadhyay, B. Gain et A. Ekbal : A transformer-based approach to multilingual fake news detection in low-resource languages. *ACM Trans. Asian Low-Resour. Lang. Inf. Process.*, 21(1):247–251, 2021.

[21] A. V. Deeb : 'fake news' has been added to the oxford english dictionary.

[22] J. Devlin, M. Chang, K. Lee et K. Toutanova : Bert: Pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805, 2018.

[23] S. Dhall, A. D. Dwivedi, S. K. Pal et G. Srivastava : Blockchain-based framework for reducing fake or vicious news spread on social media/messaging platforms. *ACM Trans. Asian Low-Resour. Lang. Inf. Process.*, 21(1), 2021.

[24] J. Fawaid, A. Awalina, R. Y.Krisnabayu et N. Yudistira : Indonesia's fake news detection using transformer network. *6th International Conference on Sustainable Information Engineering and Technology 2021*, page 247–251, 2021.

[25] L. J. Y. Flores et Y. Hao : An adversarial benchmark for fake news detection models. *CoRR*, abs/2201.00912, 2022.

[26] P. Gamallo : CiTIUS at fakedes 2021: A hybrid strategy for fake news detection. *IberLEF 2021: Iberian Languages Evaluation Forum 2021*, 2021.

[27] S. C. R. Gangireddy, Deepak P, C. Long et T. Chakraborty : Unsupervised fake news detection: A graph-based approach. *Proceedings of the 31st ACM Conference on Hypertext and Social Media*, page 75–83, 2021.

[28] M. Gôlo, M. Caravanti, R. Rossi, S. Rezende, B. Nogueira et R. Marcacini : Learning textual representations from multiple modalities to detect fake news through one-class learning. *Proceedings of the Brazilian Symposium on Multimedia and the Web*, page 197–204, 2021.

[29] M. Grandini, E. Bagli et G. Visani : Metrics for multi-class classification: an overview. *CoRR*, abs/2008.05756, 2020.

[30] N. Grinberg, K. Joseph, L. Friedland, B. Swire-Thompson et D. Lazer : Fake news on twitter during the 2016 u.s. presidential election. *Science*, 363(6425):374–378, 2019.

[31] P. Gupta, S. Gandhi et B. R. Chakravarthi : Leveraging transfer learning techniques- bert, roberta, albert and distilbert for fake review detection. *Forum for Information Retrieval Evaluation*, pages 75–82, 2022.

[32] Y. Han, A. Silva, L. Luo, S. Karunasekera et C. Leckie : Knowledge enhanced multi-modal fake news detection. *CoRR*, abs/2108.04418, 2021.

[33] F. Harrag et M. K. Djahli : Arabic fake news detection: A fact checking based deep learning approach. *ACM Trans. Asian Low-Resour. Lang. Inf. Process.*, 21(4):76–80, 2022.

[34] S. Hasan, R. Alam et M. A. Adnan : Truth or lie: Pre-emptive detection of fake news in different languages through entropy-based active learning and multi-model neural ensemble. *2020 IEEE/ACM*

International Conference on Advances in Social Networks Analysis and Mining (ASONAM), pages 55–59, 2021.

[35] H. Huang, L. Zhou et Y. Jiang : Early detection of fake news based on multiple information features. *2021 4th International Conference on Data Science and Information Technology*, pages 414–419, 2021.

[36] R. Jain, Dharana D. K. Jain et N. Sharma : Fake news classification: A quantitative research description. *ACM Trans. Asian Low-Resour. Lang. Inf. Process.*, 21(1), 2021.

[37] F. Jouyandeh, S. Sadeghi, B. Rahmatikargar et P. M. Zadeh : Fake news and covid-19 vaccination: A comparative study. *Proceedings of the 2021 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, page 525–531, 2021.

[38] A. S. Karnyoto, C. Sun, B. Liu et X. Wang : Augmentation and heterogeneous graph neural network for aaai2021-covid-19 fake news detection. *International Journal of Machine Learning and Cybernetics*, 13:2033–2043, 2022.

[39] M. Kim, S. Tariq et S. S. Woo : Cored: Generalizing fake media detection with continual representation using distillation. *2020 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*, page 337–346, 2021.

[40] S. Kumari : Nofake at checkthat!2021: Fake news detection using bert. *ArXiv*, abs/2108.05419:754–763, 2021.

[41] A. Lao, C. Shi et Y. Yang : Rumor detection with field of linear and non-linear propagation. *Proceedings of the Web Conference 2021*, page 3178–3187, 2021.

[42] X. Lei : Increment-aware dynamic propagation embedding for rumor detection. *Proceedings of the 2021 5th International Conference on Electronic Information Technology and Computer Engineering*, page 745–751, 2021.

[43] K. Li : Haha at fakedes 2021: A fake news detection method based on tf-idf and ensemble machine learning. *IberLEF@SEPLN*, 2021.

[44] Y. Liu et Y. Wu : Early detection of fake news on social media through propagation path classification with recurrent and convolutional networks. *Proceedings of the AAAI Conference on Artificial Intelligence*, 32(1), 2018.

[45] Y. Liu et Y. B. Wu : Early detection of fake news on social media through propagation path classification with recurrent and convolutional networks. *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence and Thirtieth Innovative Applications of Artificial Intelligence Conference and Eighth AAAI Symposium on Educational Advances in Artificial Intelligence*, 2021.

[46] Y. Lu et C. Li : Gcan: Graph-aware co-attention networks for explainable fake news detection on social media. *CoRR*, page 395–405, 2020.

[47] S. M. Lundberg et S. Lee : A unified approach to interpreting model predictions. *31st Conference on Neural Information Processing Systems NeurIPS 2017*, 2017.

[48] H. Luo : Yeti at fakedes 2021: Fake news detection in spanish with albert. *IberLEF@SEPLN*, 2021.

[49] Z. I. Mahid, S. Manickam et S. Karuppayah : Fake news on social media: Brief review on detection techniques. *2018 Fourth International Conference on Advances in Computing, Communication Automation (ICACCA)*, pages 1–5, 2018.

[50] D. Megías, M. Kuribayashi, A. Rosales et W. Mazurczyk : Dissimilar: Towards fake news detection using information hiding, signal processing and machine learning. *The 16th International Conference on Availability, Reliability and Security*, 2021.

[51] M. Meyers, G. Weiss et G. Spanakis : Fake news detection on twitter using propagation structures. *Disinformation in Open Online Media*, pages 138–158, 2020.

[52] A. Nagaraja, Soumya K N, A. Sinha, J. V. R. Kumar et P. Nayak : Fake news detection using machine learning methods. *International Conference on Data Science, E-Learning and Information Systems 2021*, page 185–192, 2021.

[53] J. A. Nasir, O. S. Khan et I. Varlamis : Fake news detection: A hybrid cnn-rnn based deep learning approach. *International Journal of Information Management Data Insights*, 1(1), 2021.

[54] V. Nguyen, K. Sugiyama, P. Nakov et M. Kan : Fang: Leveraging social context for fake news detection using graph representation. *Proceedings of the 29th ACM International Conference on Information Knowledge Management*, page 1165–1174, 2021.

[55] S. Ni, J. Li et H. Kao : Mvan: Multi-view attention networks for fake news detection on social media. *IEEE Access*, 9:106907–106917, 2021.

[56] M. Ott, Y. Choi, C. Cardie et J. T. Hancock : Finding deceptive opinion spam by any stretch of the imagination. *corr*, abs/1107.4557, 2011.

[57] P. Patwa, S. Sharma, S. Pykl, V. Guptha, G. Kumari, M. S. Akhtar, A. Ekbal, A. Das et T. Chakraborty : Fighting an infodemic: COVID-19 fake news dataset. *CoRR*, abs/2011.03327, 2020.

[58] A. Pritzkau : Nlytics at checkthat! 2021: Multi-class fake news detection of news articles and domain identification with roberta - a baseline model. *CLEF*, 2021.

[59] P. Przybyla : Capturing the style of fake news. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(01):490–497, 2020.

[60] P. Przybyła et A. J. Soto : When classification accuracy is not enough: Explaining news credibility assessment. *Information Processing Management*, 58(5):102653, 2021.

[61] B. Rath, A. Salecha et J. Srivastava : Detecting fake news spreaders in social networks using inductive representation learning. *2020 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*, page 182–189, 2021.

[62] S. Raza : Automatic fake news detection in political platforms - a transformer-based approach. *Proceedings of the 4th Workshop on Challenges and Applications of Automated Extraction of Socio-political Events from Text (CASE 2021)*, 2021.

[63] D. Refaeli et P. Hajek : Detecting fake online reviews using fine-tuned bert. *2021 5th International Conference on E-Business and Internet*, pages 76–80, 2022.

[64] N. Reimers et I. Gurevych : Sentence-bert: Sentence embeddings using siamese bert-networks. *EMNLP 2019*, 2019.

[65] J. C. S. Reis, A. Correia, F. Murai, A. Veloso et F. Benevenuto : Explainable machine learning for fake news detection. *Proceedings of the 10th ACM Conference on Web Science*, page 17–26, 2019.

[66] J. Reyes-Magana et L. E. Argota : Forcenlp at fakedes 2021: Analysis of text features applied to fake news detection in spanish. *IberLEF@SEPLN*, 2021.

[67] M. T. Ribeiro, S. Singh et C. Guestrin : "why should i trust you?": Explaining the predictions of any classifier. *CoRR*, abs/1602.04938, 2016.

[68] G. Ruffo, A. Semeraro, A. Giachanou et P. Rosso : Surveying the research on fake news in social media: a tale of networks and language. *CoRR*, abs/2109.07909, 2021.

[69] A. Rusli, J. C. Young et N. M. S. Iswari : Identifying fake news in indonesian via supervised binary text classification. *2020 IEEE International Conference on Industry 4.0, Artificial Intelligence, and Communications Technology (IAICT)*, pages 86–90, 2020.

[70] S. R. Sahoo et B. B. Gupta : Multiple features based approach for automatic fake news detection on social networks using deep learning. *Applied Soft Computing*, 100, 2021.

[71] A. R. Sarkar et S. Ahmad : A new approach to expert reviewer detection and product rating derivation from online experiential product reviews. *Heliyon*, 7(7), 2021.

[72] X. Schmitt, S. Kubler, J. Robert, M. Papadakis et Y. LeTraon : A replicable comparison study of ner software: Stanfordnlp, nltk, opennlp, spacy, gate. *2019 Sixth International Conference on Social Networks Analysis, Management and Security (SNAMS)*, pages 338–343, 2019.

[73] P. Sehgal, B. Bhutani, N. Rastogi, A. Sinha et M. Rathi : Ai driven identification of fake news propagation in twitter social media with geo-spatial analysis. *2021 Thirteenth International Conference on Contemporary Computing (IC3-2021)*, page 432–437, 2021.

[74] K. Sharma, F. Qian, H. Jiang, N. Ruchansky, M. Zhang et Y. Liu : Combating fake news: A survey on identification and mitigation techniques. *CoRR*, abs/1901.06437, 2019.

[75] S. Sheikhi : An effective fake news detection method using woa-xgbtree algorithm and content-based features. *Applied Soft Computing*, 109:107559, 2021.

[76] H. Shelar, G. Kaur, N. Heda et P. Agrawal : Named entity recognition approaches and their comparison for custom ner model. *Science & Technology Libraries*, 39(3):324–337, 2020.

[77] K. Shu, L. Cui, S. Wang, D. Lee et H. Liu : Defend: Explainable fake news detection. page 395–405, 2019.

[78] K. Shu, D. Mahudeswaran et H. Liu : Fakenewstracker: a tool for fake news collection, detection, and visualization. *Computational and Mathematical Organization Theory*, 25, 2019.

[79] K. Shu, D. Mahudeswaran, S. Wang, D. Lee et H. Liu : Fakenewsnet: A data repository with news content, social context and spatialtemporal information for studying fake news on social media. *CoRR*, abs/1809.01286, 2018.

[80] K. Shu, S. Wang et H. Liu : Beyond news contents: The role of social context for fake news detection. *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*, page 312–320, 2019.

[81] A. Silva, Y. Han, L. Luo, S. Karunasekera et C. Leckie : Propagation2vec: Embedding partial propagation networks for explainable fake news early detection. *Information Processing Management*, 58(5), 2021.

[82] S. Singhal, M. Dhawan, R. R. Shah et P. Kumaraguru : Inter-modality discordance for multimodal fake news detection. *ACM Multimedia Asia*, 2022.

[83] S. M. Sohan, S. A. Khusbu, S. Islam et A. Hasan : Blackops at checkthat! 2021: User profiles analyze of intelligent detection on fake tweets notebook for pan. *CLEF*, 2021.

[84] J. Sol : The long and brutal history of fake news.

[85] C. Song, K. Shu et B. Wu : Temporally evolving graph neural network for fake news detection. *Information Processing Management*, 58(6), 2021.

[86] M. A. Spalenza, L. L. Filho, F. França, P. Lima et E. d. Oliveira : Lcad - ufes at fakedes 2021: Fake news detection using named entity recognition and part-of-speech sequences. *IberLEF@SEPLN*, 2021.

[87] S. Storks, Q. Gao et J. Y. Chai : Commonsense reasoning for natural language understanding: A survey of benchmarks, resources, and approaches. *CoRR*, abs/1904.01172, 2020.

[88] R. Upadhyay, G. Pasi et M. Viviani : Health misinformation detection in web content: A structural-, content-based, and context-aware approach based on web2vec. *Proceedings of the Conference on Information Technology for Social Good*, pages 19–24, 2021.

[89] S. Vosoughi, D. Roy et S. Aral : The spread of true and false news online. *Science*, 359(6380):1146–1151, 2018.

[90] A. Wani, I. Joshi, S. Khandve, V. Wagh et R. Joshi : Evaluating deep learning approaches for covid19 fake news detection. *Combating Online Hostile Posts in Regional Languages during Emergency Situation*, pages 153–163, 2021.

[91] C. Wilson : The difference between real-time, near real-time, and batch processing in big data.

[92] K. Wu, X. Yuan et Y. Ning : Incorporating relational knowledge in explainable fake news detection. *Advances in Knowledge Discovery and Data Mining*, pages 403–415, 2021.

[93] Y. Xie, X. Huang, X. Xie et S. Jiang : A fake news detection framework using social user graph. *Proceedings of the 2020 2nd International Conference on Big Data Engineering*, page 55–61, 2021.

[94] J. Xing, S. Wang, X. Zhang et Y. Ding : Hmbi: A new hybrid deep model based on behavior information for fake news detection. *Wireless Communications and Mobile Computing*, 2021.

[95] J. Xu, V. Zadorozhny, D. Zhang et J. Grant : Fands: Fake news detection system using energy flow. *Data Knowledge Engineering*, 139, 2022.

[96] J. Xue, Y. Wang, Y. Tian, Y. Li, L. Shi et L. Wei : Detecting fake news by exploring the consistency of multimodal data. *Information Processing Management*, 58(5), 2021.

[97] Z. Zhang, X. Yi et X. Zhao : Fake speech detection using residual network with transformer encoder. *Proceedings of the 2021 ACM Workshop on Information Hiding and Multimedia Security*, pages 13–22, 2021.

[98] X. Zhou, A. Jain, V. V. Phoha et R. Zafarani : Fake news early detection: A theory-driven model. *Digital Threats*, 1(2), 2020.